AIX on Power - Performance FAQ

December 4, 2012

Dirk Michel
dirkm@us.ibm.com
IBM Corporation

Preface 7
Acknowledgements 8
1 Introduction 9
1.1 Purpose of this document 9
1.2 Overview 9
1.3 Document Responsibilities 9
2 What Is Performance? 10
2.1 Introduction 10
2.2 Performance 10
2.3 Response Time 10
2.4 Throughput 10
2.5 Throughput versus Response Time 10
2.6 Acceptable Performance 11
3 Performance Benchmarks 12
3.1 Introduction 12
3.2 What Are Performance Benchmarks? 12
3.3 SPEC Benchmarks 12
3.3.1 Introduction 12
3.3.2 SPEC CPU2006 12
3.3.3 SPECpower_ssj2008 12
3.3.4 SPECjbb2005 13
3.3.5 SPECjEnterprise2010 13
3.3.6 SPECweb2005 13
3.3.7 SPEC SFS 14
3.4 TPC Benchmarks 14
3.4.1 Introduction 14
3.4.2 TPC-C 15
3.4.3 TPC-H 16
3.4.4 TPC-E 18
3.5 HPC Benchmarks 20
3.5.1 Linpack 20
3.6 rPerf 20
3.6.1 Overlap in Performance 21
3.7 Custom Benchmarks and Proof of Concepts (PoC) 21
3.7.1 Considerations 22
3.7.2 Common Pitfalls for Custom Benchmarks 22
3.8 Comparing Benchmark Results 22
3.9 References 23
4 Workload Estimation and Sizing 24
4.1 Introduction 24
4.2 Benchmarks and Sizing 24

4.3 System Sizing Tool 24
5 Performance Concepts 26
5.1 Introduction 26
5.2 CPU Performance 26
5.2.1 Central Processing Unit (CPU) 26
5.2.1 Multi-Processor Systems 27
5.2.3 Multi Threading 28
5.2.4 Processor Virtualization 29
5.3 Memory Performance 29
5.3.1 Memory Hierarchy 29
5.3.2 Virtual Memory 31
5.3.3 Memory Affinity 32
5.4 Storage Performance 32
5.4.1 Storage Hierarchy 32
5.4.2 I/O Path 34
5.4.3 Network File System I/O Path 35
5.4.4 Storage Virtualization 36
5.5 Network Performance 37
5.5.1 Network Hierarchy 37
5.5.2 Network Virtualization 39
5.5.3 Shared Ethernet Adapter 40
5.5.4 Host Ethernet Adapter 40
5.6 Software Performance 41
5.6.1 Employ up-to-date software 41
5.6.2 Compiler Optimization 42
5.6.3 Platform specific optimization 42
5.6.4 Software testing 42
5.7 Performance Metrics 43
5.7.1 System Components Performance Metrics 43
6 Performance Analysis and Tuning Process 44
6.1 Introduction 44
6.1.1 Performance monitoring before a problem occurs 44
6.2 Defining a Performance Problem (What is slow?) 44
6.3 Top-down Performance Analysis 45
6.3.1 Application 45
6.3.2 System 46
6.3.3 System Components and Kernel 46
6.3.4 Top down performance analysis flow chart 47
7 Performance Analysis How-To 48
7.1 Introduction 48
7.2 How to tune VMM page replacement to reduce paging 48

7.3 How to address CPU bottlenecks using tprof 49
7.4 How to address paging issues 50
7.4.1 What's causing paging? 51
7.5 How to address NFS sequential read/write performance problems 52
7.5.1 What is causing slow NFS sequential read/write performance? 53
7.6 How to migrate from cached file system to concurrent I/O 53
7.6.1 Effective Cache Hit Ratio 53
7.6.2 How to determine the effective cache hit ratio 55
7.6.3 How to avoid the pitfall 55
7.7 How to tune TCP/IP for Superpacket IB interface 56
7.7.1 Recommended feature usage based on machine type 56
7.7.2 Tuning for best MTU size of the IB interfaces 57
7.7.3 Configuring ml0 to match the performance tuning on IB interfaces 57
8 Frequently Asked Questions 58
8.1 Introduction 58
8.2 General Questions 58
8.2.1 What are the general performance tuning recommendations for best AIX performance? 58
8.2.2 I heard that... should I change...? 58
8.2.3 I found a three year old best practice paper on the web, is it still valid? 58
8.2.4 Why don't I see all tunables on AIX 6.1? 58
8.2.5 Do I need to recompile my application to get good performance on a new platform? 58
8.3 CPU Questions 58
8.3.1 What is I/O wait? 58
8.3.2 Why aren't all CPUs evenly being used on a shared micro partition? 59
8.4 Memory Questions 59
8.4.1 How to tune VMM page replacement to reduce paging? 59
8.4.2 When should I enable page_steal_method? 59
8.4.3 What formula should I use to calculate minfree and maxfree? 59
8.4.4 Memory pools are unbalanced; should memory affinity be turned off? 60
8.4.5 Does 'avm' in vmstat indicate "available memory"? 60
8.4.6 Why is my free memory so low? 60
8.4.7 Should I pin my database memory using v_pinshm? 60
8.5 Disk I/O Questions 60
8.5.1 What is the recommended queue_depth for disks? 60
8.5.2 What is the recommended num_cmd_elems for FC adapters? 61
8.5.3 How to tune AIO on AIX 6.1? 61
8.5.4 How do I enable AIO on AIX 6.1? 61
8.5.5 What are good values for the file I/O pacing tunables minpout and maxpout? 61
8.5.6 Does CIO always results in better database performance? 61
8.6 Network Questions 61
8.6.1 What are the recommended values for rfc1323, tcp_sendspace, and tcp_recvspace? 61

8.6.2 When should I change rfc1323, tcp_sendspace, and tcp_recvspace? 62
8.6.3 I want to change the rfc1323, tcp_sendspace and tcp_recvspace for multiple network interfaces; should I change them globally with the no command and turn off use_isno? 62
8.6.4 What are dog threads? 62
8.6.5 When should I enable dog threads for a network interface? 62
8.6.6 When should I enable link level flow control? 63
8.6.7 What are checksum offload and large send? 63
8.6.8 When should I disable checksum offload and large send? 63
8.6.9 How can I improve the loopback performance? 64
9 POWER7 65
9.1 Introduction 65
9.2 Compatibility Mode 65
9.3 Memory Considerations 65
9.4 Single thread versus SMT2/SMT4 65
9.5 Adapter Placement 65
9.6 Affinitized partitions 65
9.7 POWER7 CPU utilization reporting 66
9.7.1 CPU utilization example for dedicated LPAR 66
9.7.2 CPU utilization example for shared LPAR 68
9.8 AIX Scheduling 69
9.8.1 Scaled Throughput Dispatching 70
9.10 Virtualization Best Practices 73
9.10.1 Sizing virtual processors 73
9.10.2 Entitlement considerations 73
9.10.3 Virtual Processor Management 73
9.10.4 Best memory and CPU resource assignment for critical LPARs 73
9.11 Virtual Ethernet Performance 74
9.11.1 Sending large data packets through “largesend” 74
9.11.2 Virtual Ethernet Adapter Buffers 75
9.11.3 Data Cache Block Flush 77
9.11.4 Shared Ethernet Adapter 77
9.12 Storage Virtualization Best Practice 78
9.12.1 VIOS Sizing and Uncapped Shared Weight 78
9.12.2 vSCSI client queue depth 78
9.12.3 vSCSI virtual adapter count 78
9.13 Performance Advisor Tools 78
10 Java 80
10.1 Introduction 80
10.2 32-bit versus 64-bit Java 80
10.3 Medium page size usage (64K pages) 80
10.4 Application Scaling 80

10.5 Enhanced Tuning with Resource Sets and Memory Affinity 81
11 IBM AIX Dynamic System Optimizer 82
11.1 Introduction 82
11.2 Overview 82
11.3 How to enable ASO/DSO 82
11.3.1 Prerequisites 82
11.3.2 Install and enable DSO 82
11.4 Performance Expectation 83
11.5 Reference 83
12 Reporting a Performance Problem 84
12.1 Introduction 84
12.2 Define the Performance Problem 84
12.3 Performance Data Collection using PERFPMR 84
12.4 PERFPMR vs. test case 85
12.5 Questions that help IBM diagnose the problem 85

Preface

This document is intended to address most frequently asked questions concerning AIX® 5.3, AIX 6.1 and AIX 7.1 performance on Power Systems, and provide best practice guidelines for most commonly seen performance issues.

The following is a list of IBM® reference and documents used:

- The Performance Management Guide is a great place to start for acquiring basic performance concepts and good cross reference when reading this document: http://pic.dhe.ibm.com/infocenter/aix/v7r1/index.jsp?topic=/com.ibm.aix.prftungd/doc/prftungd/performance_management-kickoff.htm
- The System Performance Report provide benchmark results and relative performance metrics of IBM Power Systems: http://www-03.ibm.com/systems/power/hardware/reports/system_perf.html
- The AIX documentation is to look up information on particular AIX commands, tools, etc: http://publib.boulder.ibm.com/infocenter/aix/v6r1/index.jsp
- The POWER7 Virtualization Best Practice Guide, available on developerWorks, provides detailed best practices for POWER7 virtualization: https://www.ibm.com/developerworks/wikis/download/attachments/53871915/P7_virtualization_bestpractice.doc?version=1
- The Java Performance on POWER7 document describes performance considerations, tuning options, tools and diagnostic options for Java workloads: http://www-03.ibm.com/systems/power/hardware/whitepapers/java_perf.html

Acknowledgements

We would like to thank many people who made invaluable contributions to “AIX on Power Performance”. Contributions included authoring, insights, ideas, reviews, critiques and reference documents

Our special thanks to key contributors from IBM STG Cross Platform Systems Performance:

Herman Dierks, Hong Hua, Bernie King-Smith, Kiet Lam, Ann Matzou, Augie Mena III, Klavs Pedersen, Lilian Romero, Sergio Reyes, David Navarro, Brian Twitchell, Thanh Do

Our special thanks to key contributors from AIX Development Support:

Grover Davidson, William Quinn

1 Introduction

1.1 Purpose of this document

The purpose of this document is to provide a basic understanding of AIX on Power Systems performance concepts, industry standard benchmarks on Power Systems, capacity planning, pre-sales and post-sales process, performance monitoring and analysis, frequently asked questions and guidelines addressing common performance issues.

This document is not intended to replace performance management documentation or performance white papers.

1.2 Overview

This document covers a variety of Power Systems performance including:

- What Is Performance?
- Performance Benchmarks
- Workload Estimation and Sizing
- Performance Concept
- Performance Analysis How-To
- Frequently Asked Questions
- Reporting A Performance Problem

1.3 Document Responsibilities

The IBM STG Cross Platform Systems Performance organization is responsible for editing and maintaining the AIX on Power - Performance FAQ document. Any contributions or suggestions for additions or edits should be forwarded to Dirk Michel, dirkm@us.ibm.com.

2 What Is Performance?

2.1 Introduction

The purpose of this chapter is to explain what exactly computer performance is.

2.2 Performance

Computer performance is largely determined by a combination of response time and throughput. Other aspects associated with computer performance are availability of computer systems and their power efficiency.

2.3 Response Time

The response time of a computer system is the elapsed time between the end of an inquiry or demand and beginning of a response to that inquiry. For interactive users, the response time is the time from when the users hits the <enter> button to seeing the result displayed. The response time often is seen as a critical aspect of performance because of its potential visibility to end users or customers.

Let's take the example of a computer system that is running a web server with an online store. The response time here is the elapsed time between pressing the submit button to place an order and the beginning of receiving the order confirmation.

2.4 Throughput

The throughput of a computer system is a measure of the amount of work performed by a computer system over the period of time. Examples for throughput are megabytes per second read from a disk, database transactions per minute, megabytes transmitted per second through a network adapter.

Let's go back to the previous example with the online store. The computer system on which the online store is running might be one out of many computers that are all connected to the same database server. While response time of the database server is still an important factor, its throughput is more important since it processes many requests from the web servers in parallel.

2.5 Throughput versus Response Time

Throughput and response time are related. In many cases a higher throughput comes at the cost of poorer response or slower response as well as better response time comes at the cost of lower throughput.

Let's assume you load a truck with 10,000 1TB disks, drive the truck 30 miles and unload the disks within one hour. The throughput would be 2.8TB per second but at the cost of the response time which would be an hour.

Now take a muscle car instead of a truck. It's unlikely that 10,000 disks would fit into a small car, so let's assume we can load 100 1TB disks, drive the muscle car 30 miles and unload the

disks within 20 minutes. The response time now is three times better compared to the truck, however, the throughput reduced to 0.083TB per second.

2.6 Acceptable Performance

It is best to evaluate performance of a system through objective measurements, for example through application log files, or batch job run times.

Acceptable performance is based on customer expectations. Expectations can be based on benchmarks, modeling, or experience. Incorrect assumptions when architecting a system may create a situation where acceptable performance cannot be attained.

3 Performance Benchmarks

3.1 Introduction

This chapter covers the performance benchmarks and how they should be used to compare system performance. It covers the most commonly used industry standard and known industry standard benchmarks on Power Systems including SPEC, TPC and rPerf.

3.2 What Are Performance Benchmarks?

Performance benchmarks are well defined problems or tests that serve as a basis to evaluate and compare the performance of computer systems. Performance benchmark tests use representative sets of programs and data designed to evaluate the performance of computer hardware and software in a given configuration.

3.3 SPEC Benchmarks

3.3.1 Introduction

This section provides an overview of a subset of the Standard Performance Evaluation Corporation (SPEC) benchmarks. SPEC provides a standardized set of benchmarks to evaluate the performance of the newest generation of high-performance computers. For a complete list of benchmarks and corresponding descriptions, please visit http://www.spec.org.

3.3.2 SPEC CPU2006

SPEC CPU2006 is an industry-standard benchmark designed to provide performance measurements that can be used to compare compute-intensive workloads on different computer systems, SPEC CPU2006 contains two benchmark suites: CINT2006 for measuring and comparing compute-intensive integer performance, and CFP2006 for measuring and comparing compute-intensive floating point performance.

For more information, please see http://www.spec.org/cpu2006/

3.3.3 SPECpower_ssj2008

SPECpower_ssj2008 is the first industry-standard SPEC benchmark that evaluates the power and performance characteristics of volume server class computers.

SPEC has designed SPECpower_ssj2008 to be used as both a benchmark to compare power and performance among different servers and as a toolset to improve server efficiency.

The benchmark workload represents typical server-side Java business applications. The workload is scalable, multi-threaded, portable across a wide range of operating environments, and economical to run. It exercises the CPUs, caches, memory hierarchy and the scalability of shared memory processors (SMPs), as well as the implementations of the JVM (Java Virtual Machine), JIT (Just-In-Time) compiler, garbage collection, threads and some aspects of the operating system.

The metric for SPECpower_ssj2008 is “overall ssj_ops/watt”.

For more information, please see http://www.spec.org/power_ssj2008/

3.3.4 SPECjbb2005

SPECjbb2005 is an industry-standard benchmark designed to measure the server-side performance of Java runtime environment.

The benchmark evaluates the performance of server side Java by emulating a three-tier client/server system with emphasis on the middle tier. It exercises the implementations of the JVM (Java Virtual Machine), JIT (Just-In-Time) compiler, garbage collection, threads and some aspects of the operating system. It also measures the performance of CPUS, caches, memory hierarchy and the scalability of shared memory processors (SMPs).

The metrics for this benchmark is SPECjbb2005 bops (business operations per second), and SPECjbb2005 bops/JVM.

For more information, please see http://www.spec.org/jbb2005/

3.3.5 SPECjEnterprise2010

SPECjEnterprise2010 is a multi-tier benchmark for measuring the performance of Java Enterprise Edition 5.0 (Java EE 5.0) technology based application server.

The benchmark stresses all major Java EE 5.0 technologies implemented by compliant Java EE 5.0 application servers. It also heavily exercises all parts of the underlying infrastructure that make up the application environment, including hardware, JVM software, database software, JDBC drivers, and the system network.

The metric for SPECjEnterprise2010 is Enterprise jAppServer Operations Per Second (“SPECjEnterprise2010 EjOPS”).

For more information, please see http://www.spec.org/jEnterprise2010/

3.3.6 SPECweb2005

The SPECweb2005 benchmark includes workloads to measure banking, e-commerce and support web server performance using HTTP (non-secure), HTTPS (secure) and a mix of secure and non-secure HTTP connections.

SPECweb2005 supersedes the SPECweb99 and SPECweb99_SSL benchmarks which were retired in October 2005. It includes enhancements like dynamic web content, simulation of browser caching and the ability to poll clients for data during the runs.

For more information, please see http://www.spec.org/web2005/

3.3.7 SPEC SFS

The SPECsfs97_R1 benchmark includes workloads to measure both NFS V2 and NFS V3 server performance over UDP and TCP. Due to NFS V2 and UDP becoming less prevalent in customer environments, the primary workload receiving most focus is SPECsfs97_R1.v3 over TCP. The metrics for this benchmark include peak throughput (in NFS ops/sec) and response time (in msec/op).

The SPECsfs2008 benchmark supersedes SPECsfs97_R1. It contains an updated NFS V3 workload, plus a workload to measure Common Internet File System (CIFS) server performance. Among other changes, the NFS V3 workload was modified to reflect industry trends of increasing file sizes and increasing sizes of over-the-wire Remote Procedure Calls (RPCs).

SPEC SFS stresses many kernel subsystems, including NFS (or CIFS), Virtual Memory Management subsystem (VMM), Process and Threads Management subsystem (sysproc), the native file system being exported (e.g., JFS2), LVM, TCP/IP, storage devices drivers, network device drivers, etc. It consumes 100% system (and no user) time.

One drawback of the benchmark is the amount of auxiliary equipment required, including client machines and a significant amount of disk storage. However, SPEC SFS is an excellent benchmark for measuring overall kernel performance.

For more information, please see http://www.spec.org/sfs2008/

3.4 TPC Benchmarks

3.4.1 Introduction

This section provides a general description of the Transaction Processing Council (TPC) benchmarks. The purpose of these database benchmarks is to provide performance data to the industry.

All the TPC results must comply with standard TPC disclosure policies and be reviewed by a TPC auditor. A Full Disclosure Report and Executive Summary must be submitted to the TPC before a result can be announced.

For further detail on the TPC benchmarks and announced results refer to the TPC website: www.tpc.org

Results should be sorted by Number of Cores or even Number of Hardware Threads, since #Processors can be deceiving when comparing dual-cores with quad-cores.

3.4.2 TPC-C

The TPC-C benchmark is emulating a moderately complex on-line transaction processing (OLTP) environment. It simulates a wholesale supplier with a number of geographically distributed sales districts and associated warehouses, managing orders where a population of users executes transactions against a database.

The workload consists of five different types of transactions:

- New order: Enters a new order from a customer
- Payment: Updates customer's balance (recording payment)
- Order status: Retrieves the status of customer's most recent orders
- Delivery: Deliver orders (queued for deferred execution)
- Stock level: Monitors the stock (inventory) level

The terminal emulators execute transactions against a database consisting of nine tables and do operations like "update", "insert", "delete" and "abort". Each transaction has a response time requirement. At least 90% of each transaction must have a response time of less or equal to 5 seconds, except for stock level which is less or equal to 20 seconds.

The minimum transaction mix, the minimum keying time, a 90% response time constraint, and minimum mean of think-time distribution that must be maintained during the measurement interval is as follows:

Transaction Type	Minimum % of mix	Min Keying Time (sec)	90% Percentile Response Time Constraint (sec)	Minimum Mean of Think Time Distribution (sec)
New Order	No min – measured rate is the reported throughput	18	5	12
Payment	43%	3	5	12
Order Status	4%	2	5	10
Delivery	4%	2	5	5
Stock	4%	2	20	5

The throughput of the TPC-C workload is driven by the activity of the terminals connected to each warehouse. The users and database scale linearly with throughput.

TPC-C requires that the database transactions be ACID (Atomicity, Consistency, Isolation, and Durability) compliant.

- Atomicity: Verify that all changes within a transaction commit/abort.

- Consistency: Verify the level of database consistency across the mix of transactions.
- Isolation: ANSI repeatable reads for all but stock level transactions. The only exception is to allow non-repeatable reads (cursor stability) to stock level transaction.
- Durability: Must demonstrate recovery loss of power, memory and media.

TPC-C requires that all data partitioning be fully transparent to the application code. It allows both horizontal and vertical partitioning.

The benchmark must be conducted at steady state (sustainable throughput for a continuous period) and the measured interval must be equal or greater than 2 hours.

The primary metrics are:

- The TPC-C Maximum Qualified Throughput (MQTh) rating expressed in new-order transactions executed per minute (tpmC) which is also known as the performance metric.
- The total 3-year pricing divided by the MQTh and expressed as price/tpmC which is also known as the price/performance metric.

When comparing results one has to decide on the range of performance and price/performance that is most important and the relative importance of each factor. Typically, users start their evaluation process by stating that they are interested in systems that offer, say, 5,000,000-6,100,000 tpmC, in a price/performance range of $2-3. Once they have extracted those systems from the TPC-C's results listing, they can proceed with a more thorough investigation of the merits of each system.

3.4.3 TPC-H

The TPC-H benchmark models a decision support system by executing ad-hoc queries and concurrent updates against a standard database under controlled conditions. The purpose of the benchmark is to “provide relevant, objective performance data to industry users” according to the specifications and all implementations of the benchmark, in addition to adhering to the specifications, must be relevant to real-world (i.e. customer) implementations. TPC-H represents information analysis of an industry which must manage, sell or distribute a product worldwide. The 22 queries answer questions in areas such as pricing and promotions, supply and demand management, profit and revenue management, customer satisfaction, market share, shipping management. The refresh functions are not meant to represent concurrent on-line transaction processing (OLTP); they are meant to reflect the need to periodically update the database.

TPC-H enforces the ad-hoc model by severely restricting the implementation of auxiliary data structures such as indices and materialized query tables (sometimes known as automatic summary tables or materialized views). It also restricts how horizontal partitioning (by row) may be implemented. The partitioning column is constrained to primary keys, foreign keys and date columns. If range partitioning is used, the ranges must be divided equally between the minimum and maximum value. By imposing these restrictions, the TPC-H benchmark maintains the server platform as part of the performance equation and represents an ad-hoc environment.

The TPC-H database size is determined by the scale factor (SF). A scale factor of 1 represents a database with 10,000 suppliers and corresponds approximately to 1GB of raw data. Only a subset of scale factors are permitted for publication: 1, 10, 30, 100, 300, 1000, 3000, 10000, 30000 and 100000. The database is populated with a TPC supplied data generation program, dbgen, which creates the synthetic data set. The set of rows to be inserted or deleted by each execution of the update functions is also generated by using dbgen. The database consists of eight tables:

TABLE NAME	CARDINALITY
REGION	5
NATION	25
SUPPLIER	SF*10K
CUSTOMER	SF*150K
PART	SF*200K
PARTSUPP	SF*800K
ORDER	SF*1500K
LINEITEM	SF*6000K (approximate)

The benchmark results, which consist of two performance metrics and one price/performance metric:

- Composite Metric (QphH@Size™) = $\sqrt{QppH@size * QthH@size}$. This metric is the primary performance metric which is composed of the two pieces:
  - Power (QppH@Size™) = $\frac{3600 * SF}{\sqrt[24]{Q1 * Q2 * ... * Q22 * RF1 * RF2}}$ where Q1, Q2... RF1, RF2 are timing intervals in seconds of queries and updates. The geometric mean of the queries and updates is used here to give equal "weighting" to all the queries even though some may be much longer running than others. The power metric is derived from a power run (single stream) in which all queries and update functions are run in a specified sequence.
  - Throughput (QthH@Size™) = $\frac{NumberOfStreams * 24 * 3600 * SF}{TotalElapsedTime}$ where each stream is defined as a set of the 22 queries and 2 updates in the predefined order and total elapsed time includes the timing interval for the completion of all query streams and the parallel update stream. The throughput metric must be derived from a throughput run (multi-stream).
- Price/Performance (Price-per-QphH@size™) = $\frac{\$}{QphH@Size}$ where $ is the total hardware, software and 3 year maintenance costs for the system under test.

In addition to these TPC metrics, the number of streams is reported, which gives an indication of the amount of concurrency during the throughput run. The database load time (defined as the total elapsed time to create the tables, load data, create indices, define and validate constraints, gather database statistics and configure the system under test) is also reported.

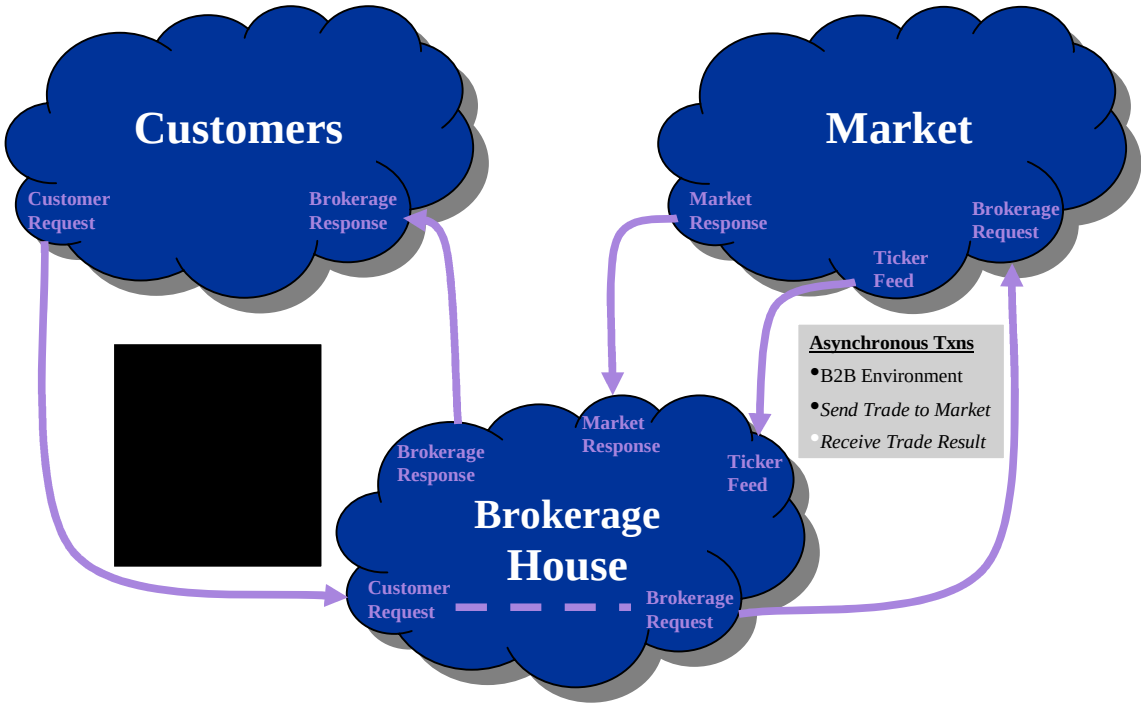
TPC-H benchmark measures the server's CPU, I/O and memory capabilities via various database operations such as full table scans, sorting, joins and aggregation. It also measures how well a RDBMS performs these basic database operations, and rewards those with efficient code paths and advanced query optimizers and parallel technology.

The Power test includes the sequential execution of 22 queries. Since each query has a different resource requirement the system utilization varies significantly during the test. It is important to understand the resource needs of the various queries and tune the system to accommodate as many queries as possible for the best query response time. The Throughput test measures the system throughput time with multiple streams (users).

The performance (Composite Metric, the higher number the better) and price/performance (price per QphH@size , the lower price the better) are the two values one would look for to understand how well the system is performed and its relative cost. And users can take the system configuration used for the benchmark to achieve an optimal, balanced system for data warehouse or business intelligent type of environment.

3.4.4 TPC-E

TPC-E is an OLTP workload designed to simulate a Stock Brokerage House, i.e. Customer trading accounts interacting with an Artificial Market Emulator.



C2B is customer to Brokerage, B2B is Brokerage to Brokerage.

The customer emulators execute a predefined mix of 11 transactions against the database all reflecting aspects of stock trading and with a preset light/heavy (processing power required) and read/write expectation,

Transaction types	Mix %	Max Response time	Weight	Access	Category	Frames
Trade-Order	10.1	2s	Heavy	rw	customer	6
Trade-Result	10.0	2s	Heavy	rw	market	6
Trade-Lookup	8.0	3s	Medium	ro	customer	4
Trade-Update	2.0	3s	Medium	rw	customer	3
Trade-Status	20.0	1s	Light	ro	customer	1
Customer-Position	14.0	3s	Mid-heavy	ro	customer	3
Broker-Volume	0.9	3s	Mid-heavy	ro	customer	1
Security-Detail	15.0	3s	Medium	ro	customer	1
Market-Feed	1.0	2s	Medium	rw	market	1
Market-Watch	19.0	3s	Medium	ro	customer	1
Data Maintenance	x per 12min	n/a	Light	rw	time	1

rw is read-write, ro is read-only, Weight is Processing power, Category indicate which part initiates transaction, Frames indicate the number of client/server interactions.

The overall metrics are called tpsE and is TradeResults per second and $/tpsE, which is the pricing component. The general scale factor is 500, which means that each 500 customers should generate within 2% of 1 tpsE to be valid. Many of the metrics have parallels to the TPC-C benchmark, however since the measurement setup does not require a full 3-tier'ed RTE (Remote Terminal Emulator) , metrics like Menu time, Keying and Think time are non-existent, only 90% percentile Response Times are reported. Apart from the overall transaction mix, many transactions have additional constraints on input values to force a certain preset diversity, for instance Customer Position must have 50% by_tax_id and 50% get_history.

A number of other workload and hardware configuration parameters are also used to categorize a particular platform

- Customers (which is an indicator of database size)
- Processor type and speed, caches, # Processor/Cores/Threads
- Physical memory
- # Physical Disks and sizes
- # Disk controller, types
- Network speed
- Server OS
- Database vendor

Generally TPC-E is a less i/o intensive, less memory requiring workload compared to TPC-C, but puts a higher demand on processing power and processor<->memory and cache activity.

3.5 HPC Benchmarks

3.5.1 Linpack

Linpack is a numerically intensive benchmark that is widely used in High Performance Computing (HPC) environments to measure floating point performance of computers. It is used to rank the top 500 supercomputer systems.

3.6 rPerf

Workloads have shifted over the last eight years and IBM is committed to providing clients with a relative system performance metric that reflects those changes. IBM publishes the rPerf relative performance metric for the IBM Power Systems family of UNIX servers. This metric replaced ROLTP which was withdrawn. rPerf is a combination of several different measures of total systems commercial performance that takes into account the demands on a server in today's environment. Although you might find historical references to ROLTP on this Web site, it will no longer be published for any currently marketed or new System p servers.

rPerf (Relative Performance) - An estimate of commercial processing performance relative to other IBM UNIX systems. It is derived from an IBM analytical model which uses characteristics from IBM internal workloads, TPC and SPEC benchmarks. The rPerf model is not intended to represent any specific public benchmark results and should not be reasonably used in that way. The model simulates some of the system operations such as CPU, cache and memory. However, the model does not simulate disk or network I/O operations.

rPerf estimates are calculated based on systems with the latest levels of AIX® and other pertinent software at the time of system announcement. Actual performance will vary based on application and configuration details. The IBM eServer™ pSeries® 640 is the baseline reference system and has a value of 1.0. Although rPerf may be used to compare estimated IBM UNIX commercial processing performance, actual system performance may vary and is dependent upon many factors including system hardware configuration and software design and configuration. Note that the rPerf methodology used for the POWER6™ systems is identical to that used for the POWER5™ systems. Variations in incremental system performance may be observed in commercial workloads due to changes in the underlying system architecture.

All performance estimates are provided "AS IS" and no warranties or guarantees are expressed or implied by IBM. Buyers should consult other sources of information, including system benchmarks, and application sizing guides to evaluate the performance of a system they are considering buying. For additional information about rPerf, contact your local IBM office or IBM authorized reseller.

3.6.1 Overlap in Performance

Figure 3.6.1 illustrates how IBM maintains your mileage may vary when looking at rPerf results. To illustrate this here would be IBM's view on how the different models could well overlap --- it is key that rPerf does include the uplift for SMT

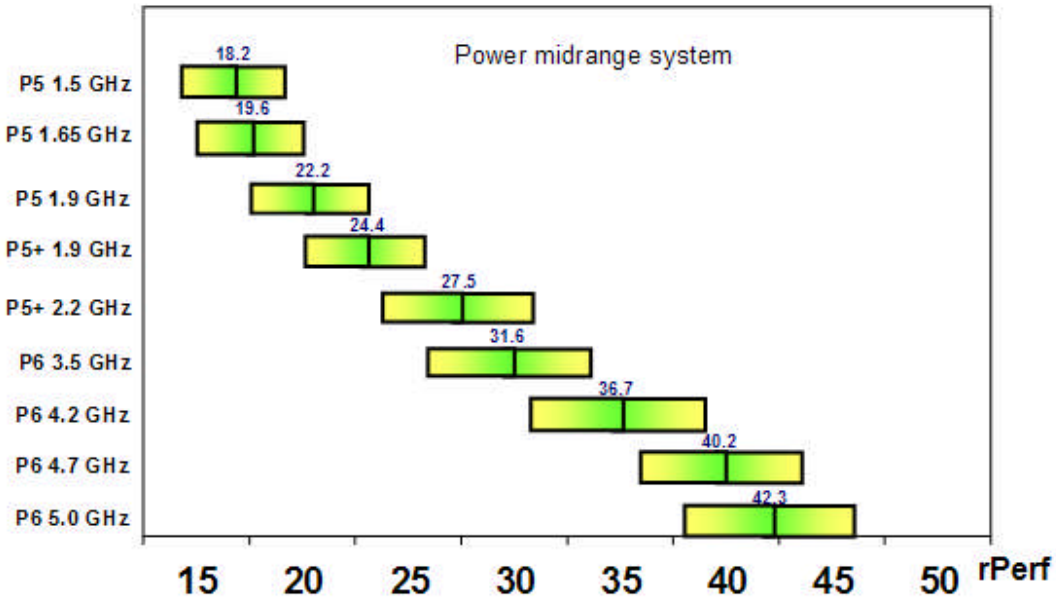


Figure 3.6.1 Overlap in Performance

3.7 Custom Benchmarks and Proof of Concepts (PoC)

Custom benchmarks are often used when industry standard benchmarks are not available for a specific computer system or its configuration or, when none of the industry standard benchmarks reflect the actual workload.

Custom benchmarks can be as simple as testing a specific function or subroutine, also known as atomic test, or as complex as recreating a multi tier application and database server environment.

When a custom benchmark is performed to measure the performance of a computer system for a customer production workload it is important that the benchmark test represents the real workload to get meaningful data.

For example, running a single database job at a time on an otherwise idle database server provides good information about the performance of the database server under best possible conditions. However, it does not provide any information about the performance of this database job when the server is under medium or heavy production workload.

3.7.1 Considerations

When doing a custom benchmark or Proof of Concept (PoC) it is important that the test be constructed to simulate the production environment. This is especially true as the hardware continues to evolve into the multi-core era and more time is being invested in the cache/memory hierarchy.

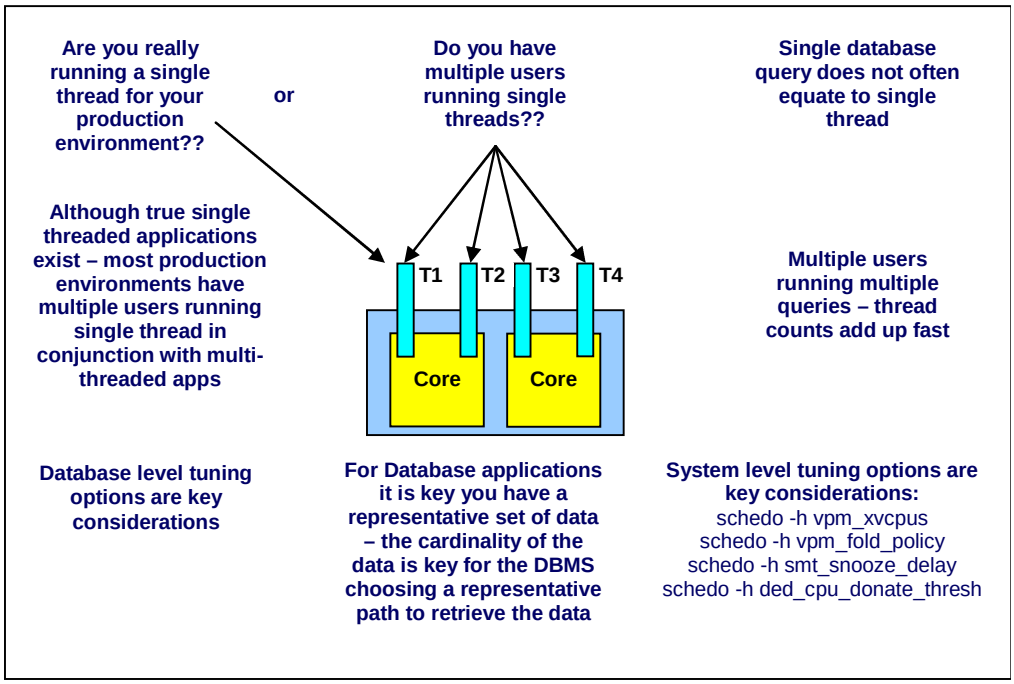


Figure 3.7.1 Considerations

3.7.2 Common Pitfalls for Custom Benchmarks

The most common pitfall when running a custom benchmark or a proof of concept is that the benchmark test does not simulate the real production environment and the benchmark result does not represent the performance the system will achieve in the production environment. The achieved benchmark result might be much better for the benchmark test than for the real production workload which most likely would lead to performance problems later on when running the real workload. It also could be the other way around potentially causing delays or failure of the PoC.

3.8 Comparing Benchmark Results

When comparing performance benchmark results it is important to compare “apples-to-apples” by comparing the results of the same performance benchmark test(s). The result of one benchmark test often does not represent the performance of a computer system for another workload.

For example, the result of a floating point intensive benchmark test doesn’t provide any information about the performance of the same computer running an integer intensive benchmark or an OLTP workload and vice versa.

A common pitfall in setting wrong performance expectations is to look at the results of one performance benchmark and apply it to another workload. An example for this would be to compare the benchmark results of two computer systems running an OLTP workload which shows that machine A is 50% faster than machine B and expect that machine A would also be 50% faster for a workload that wasn't measured.

3.9 References

TPC http://www.tpc.org

SPEC http://www.spec.org

LINPACK http://www.netlib.org/benchmark/performance.pdf

rPerf http://www-03.ibm.com/systems/p/hardware/notices/rperf.html

TOP500 http://www.top500.org/

4 Workload Estimation and Sizing

4.1 Introduction

Sizing a system, and all its various components, so that it is capable of adequately supporting a production environment can be quite complex. It requires a good knowledge of the characteristics of the workload(s) to be run, and the load that they will place on the system components.

Some questions to consider before beginning the sizing exercise:

1. What are the primary metrics, e.g., throughput, latency, that will be used to validate that the system is meeting performance requirements?
2. Does the workload run at a fairly steady state, or is it bursty, thereby causing spikes in load on certain system components? Are there specific criteria, e.g., maximum response time that must be met during the peak loads?
3. What are the average and maximum loads that need to be supported on the various system components, e.g., CPU, memory, network, storage?

4.2 Benchmarks and Sizing

It's important to understand that benchmarks typically use the latest software stack to exploit possible improvements that comes with new hardware. For example, AIX 6.1 and later take full advantage of the four hardware threads (SMT4) that where introduced with POWER7. AIX 5.3, also supported on POWER7, does not take advantage of SMT4 and therefore might not provide the same performance as the newer versions of the operating system.

Benchmarks typically provide a metric like number of transactions per time period for a system or a core. Such a metric does not provide a good base to estimate CPU consumption (utilization). For example, let's assume that system A has a benchmark result for a compute intensive workload that is 2x the capacity of system B. This could lead to the assumption that the CPU utilization of system A would be 50% when running system B's workload on it. This may be the case but there is no guarantee for it.

4.3 System Sizing Tool

Workload Estimator (WLE) is a web-based sizing tool that can be used to size a System. It takes workload characteristics as input, and provides output consisting of recommendations for the appropriate processor, memory, I/O adapter, I/O subsystem, and storage configuration for a system that will handle the workload with adequate performance.

The tool is updated frequently with new data based on actual internal workload measurements that stress key subsystem components in order to provide more accurate sizing. It can be used to size not only dedicated processor configurations, but logical partitioning and micro-partitioning environments as well.

Rather than re-inventing the wheel with spreadsheet models, etc., it is recommended that you start with WLE to see if it meets your sizing requirements.

WLE can be found at:

http://www-304.ibm.com/systems/support/tools/estimator/index.html

5 Performance Concepts

5.1 Introduction

This chapter provides a high level overview of key performance concepts.

5.2 CPU Performance

5.2.1 Central Processing Unit (CPU)

The Central Processing Unit takes a central role in the performance of a computer system. This section describes the basic characteristics of a CPU that have an impact on performance.

CPU clock speed

The clock speed is the frequency at which a CPU operates. The clock speed is in units of megahertz (MHz) on older CPUs and gigahertz (GHz) on modern CPUs. A clock speed of one megahertz is equal to one million CPU cycles and a clock speed of one gigahertz is equal to one billion CPU cycles. A CPU cycle is the interval of time needed to perform one operation - modern CPUs can perform multiple operations during one CPU cycle. A CPU cycle is not equal to a CPU instruction which often requires multiple CPU cycles to complete.

CPU Instruction

A CPU instruction is a machine or assembly language instruction that specifies the operation given to a CPU. Examples for CPU instructions are load, add, and store. Some CPU instructions are more complex than others and therefore require more CPU cycles to complete.

For example, while a CPU instruction to bit-shift the contents of a register left may take one CPU cycle, a square root instruction needs N cycles.

Cycles per Instruction

The number of CPU cycles to complete an instruction depends on the complexity of the instruction itself, whether data is available or needs to be loaded from cache or memory, and the availability of the processor units required for the instruction.

Path Length

The path length is the number of instructions it takes to complete a certain task

Execution Time

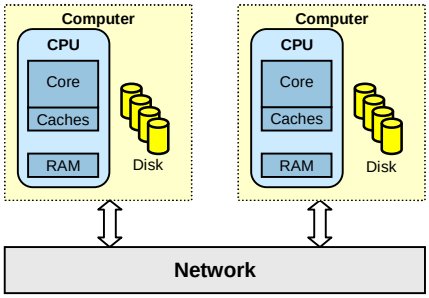
The execution time is the combination of path length, CPI and clock frequency. The execution time can be heavily impacted by the cycles per instructions.

For example, an application running on the same machine will be 20% slower when executing with a CPI of 6 relative to a CPI of 5 (6 = 1.2 *5) but 16% faster when executing with a CPI of 5 compared to a CPI of 6 (1 - 5 CPI/6 CPI) = (1-5/6) = 0.16.

5.2.1 Multi-Processor Systems

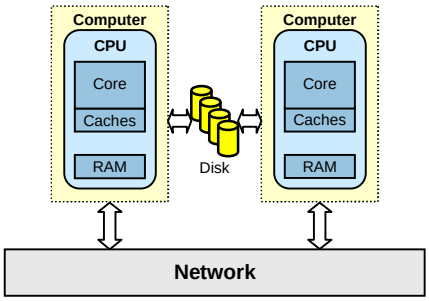
Multi processors systems are more complex than single processor systems because access to shared resources like memory needs to be serialized and the data needs to be kept synchronized across the caches of the individual CPUs.

Types of Multi-Processor Systems



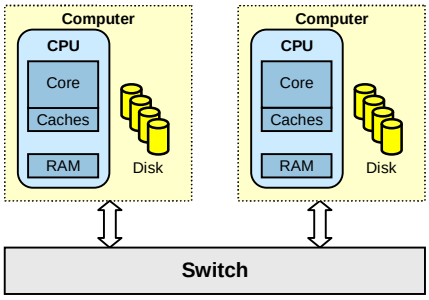
Shared Nothing MP (pure cluster)

- Each processor is a stand-alone machine
- Own copy of the operating system
- No resources shared
- Communication through network



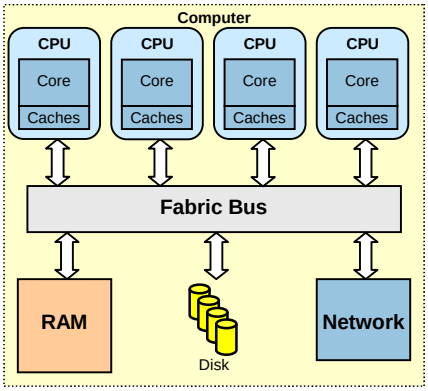
Share Disk MP

- Each processor has its own memory and cache
- Own copy of the operating system
- Processors run in parallel
- Share disks
- Communication through network



Shared Memory Cluster (SMC)

- All processors in a shared memory cluster
- Each processor has its own resources
- Own copy of the operating system
- Processor tightly coupled
- Connected through switch



Shared Memory MP

- All processors tightly coupled; all inside the same box with high speed bus or switch
- Processors share memory, disks, and I/O devices
- One copy of the operating system
- Multithreaded operating system

Figure 5.2.1 Types of Multi-Processor Systems

5.2.3 Multi Threading

Hardware multithreading originally was introduced with the models M80 and p680. Newer processors, like POWER5, POWER6 and POWER7™, support Simultaneous Multithreading, SMT, which allows both hardware threads to execute instructions at the same time. The POWER5 and POWER6 cores support single thread mode (ST) and simultaneous multithreading with two SMT threads (SMT2). Each SMT thread is represented as a logical CPU in AIX. When running in SMT2 mode, a system with a single POWER5 or POWER6 core will have two logical CPUs. The POWER7 core supports single thread mode (ST) and simultaneous multithreading with two SMT threads (SMT2) and four SMT threads (SMT4). When running in SMT4 mode, a system with a single POWER7 core will have four logical CPUs. To fully benefit from the throughput improvement of SMT, applications need to utilize all of the SMT threads of the processors.

Core	SMT Mode	Number of logical CPUs
POWER5	ST	1
POWER5	SMT2	2
POWER6	ST	1
POWER6	SMT2	2
POWER7	ST	1
POWER7	SMT2	2
POWER7	SMT4	4

The table above shows the number of logical CPUs per core when running in different SMT modes.

Cache misses can delay the execution of instructions in a processor for many cycles during which no other instruction can be executed. Multithreading address this issue by simultaneously holding the state of two or more threads. When one thread becomes stalled, due to a cache miss for example, the processor switches to the state and attempts to execute the instructions of another thread.

5.2.4 Processor Virtualization

In a virtualized environment physical processors are represented as virtual processors that can be shared across multiple partitions. The Hypervisor assigns physical processors to shared partitions, also known as SPLPAR or micro partitions, based on the capacity configurations and resource consumption of the partitions.

Virtual Processor Management

Virtual processor management, also known as Processor Folding, dynamically increases and reduces the number of virtual processors of a shared partition based on the instantaneous load of the partition. Many workloads benefit from virtual processor management due to higher degree of processor affinity.

5.3 Memory Performance

5.3.1 Memory Hierarchy

The memory of a computer system is divided into several layers of different speeds and sizes. Typically, the faster the memory is the more expensive it is to implement and therefore smaller in size.

The following figure is a high level representation of the memory hierarchy based on the location of the memory in a computer system. The implementation of the individual memory layer may differ depending on the implementation of a specific architecture.

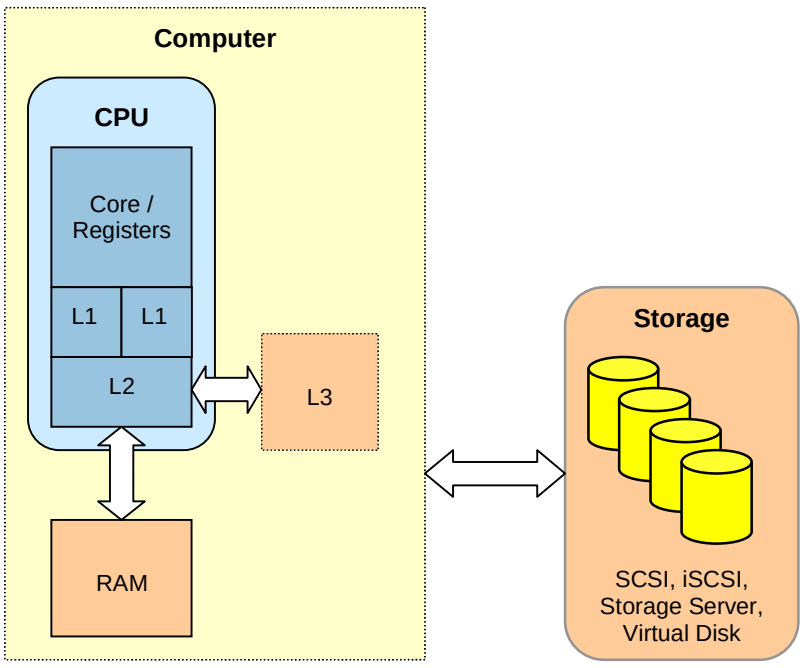


Figure 5.2.1 Memory Hierarchy

Registers

The registers of a CPU is the fastest memory and is the top layer of the memory hierarchy of a computer system. A CPU has a limited number of registers which can be used for integer and floating point operations.

Caches

Modern processors have multiple layers of caches. The fastest cache that is closest to the registers is the Level 1 cache. It is the smallest cache and often divided into a Level 1 instruction and Level 1 data cache.

The next level of cache is the Level 2 cache which often holds instructions and data. The Level 2 cache has higher access latency than the Level 1 cache but has the advantage that it can be several megabytes in size.

Some processors have a third level of cache which either can be on the same chip as the processor or external; in the latter case the processor will have a cache controller.

Cache Coherency

Cache coherency becomes an important factor on Symmetrical Multi-Processor (SMP) systems when each processor has its own cache. A coherency problem can occur when two or more processor can have a copy of the same data in their caches. To keep the data consistent, processors use snooping logic to broadcast a message over the bus each time its cache has been modified. When a processor receives a message from another processor and detects that another processor changed a value for an address that exists in its own cache, it invalidates its own copy of the data (called cross invalidate).

Cross invalidate and snooping have an impact on performance and scalability of SMT systems due to increased number of cache missed and increased bus traffic.

Random Access Memory (RAM)

The next level in the memory hierarchy is the Random Access Memory (RAM). It is much slower than the caches but also much cheaper to produce. The size of the RAM in a computer system can vary from several hundred megabytes on a small work station to several terabytes on high end servers. A processor accesses RAM either through integrated memory controllers or through bus systems which connects it to an external memory controller.

Storage

Storage is the bottom layer of the memory hierarchy since it is significant slower than any other layer. While accessing caches takes only a few CPU cycles, accessing storage can take millions of cycles.

Storage is mainly used for storing persistent data, either files in a file system or raw data used by databases. However, it is also used to temporarily store computational memory in the cases

where a computer system has insufficient real memory (RAM) or its tuning is causing unnecessary paging activity.

5.3.2 Virtual Memory

Virtual memory is a method that allows the operating system to address more memory than a computer system has real memory. Virtual memory consists of real memory and physical disk space used for working storage and file pages. On AIX, virtual memory is managed by the Virtual Memory Manager (VMM).

VMM Virtual Segments

The AIX virtual memory is partitioned into virtual segment. Each virtual segment is a continuous address space of 256 MB (default segment size) or 1TB (super segment) and further divided into pages. Pages can have multiple sizes and are not necessarily contiguous in physical memory.

The type of the VMM virtual segment defines for what type of pages the segment is being used for, i.e. working pages or file pages. The following table lists the most commonly used VMM virtual segment types.

Segment Type	Used for
Computational	Process private segments Shared segments Paging space
Client	Enhanced JFS (JFS2) file and executables NFS files and executables CD-ROM, DVD file system Compressed JFS files and executables
Persistent	JFS files and executables

Real memory

Real memory is divided into page frames which size depends on the version of AIX and the platform on which it is running. On legacy systems that don't support page sizes larger than 4 KB the real memory is divided into 4 KB frames. Platforms and AIX versions that do support larger page sizes divide the memory into frames with multiple page sizes. The following table shows what pages sizes are supported by the individual platforms and versions of AIX:

Page Size	Platform	AIX version
4 KB - small	All	3.1 and later
64 KB - medium	POWER5+ and later	5.3 and later
16 MB - large	POWER4 and later	5.1 and later
16 GB - huge	POWER6 and later	5.3 and later

AIX 5.3 and later dynamically manages pools of 4KB and 64KB page sizes. Starting with AIX 6.1 on POWER6 individual segments can have 4KB and 64KB page sizes.

Address Translation

Applications that are running on AIX, 32-bit or 64-bit, will have their own address space starting from address 0 to the highest possible address. Shared segments, the shared library segment for example, are mapped into the address space of the application.

When an application accesses memory, the effective address used by the application will be translated into a real memory address. The effective-to-real-address translation is done by the processor which maintains an effective-to-real-address (ERAT) translation table. When a processor does not have the necessary information for the address translation it will try to walk the page frame table and/or access the translation lookaside buffer (TLB) first.

5.3.3 Memory Affinity

Memory Affinity

Memory affinity is an approach to allocate memory that is closest to the processor on which a process caused a page fault. The AIX memory affinity support allows user memory allocation in a first-touch or round-robin (default) scheduling policy. The scheduling policy can be specified for individual memory types, such as data, mapped files, shared memory, stack, text and unmapped files.

An efficient use of memory affinity requires an appropriate degree of processor affinity to assure that application threads that are interrupted are redispatched to the processor(s) from which their memory was allocated.

Processor Affinity

The goal of processor affinity is to reduce the number of cache misses by redispatching an interrupted thread to the same processor it previously was running on. The efficiency of processor affinity mainly depends on contents of the processors cache. In the best case, the processor’s cache contains sufficient data from the thread and the thread’s execution can continue without any waits to resolve cache misses. In the worst case the processor’s cache has been depleted and the thread will experience a series of cache misses.

5.4 Storage Performance

5.4.1 Storage Hierarchy

Figure 5.4.1 demonstrates the storage hierarchy

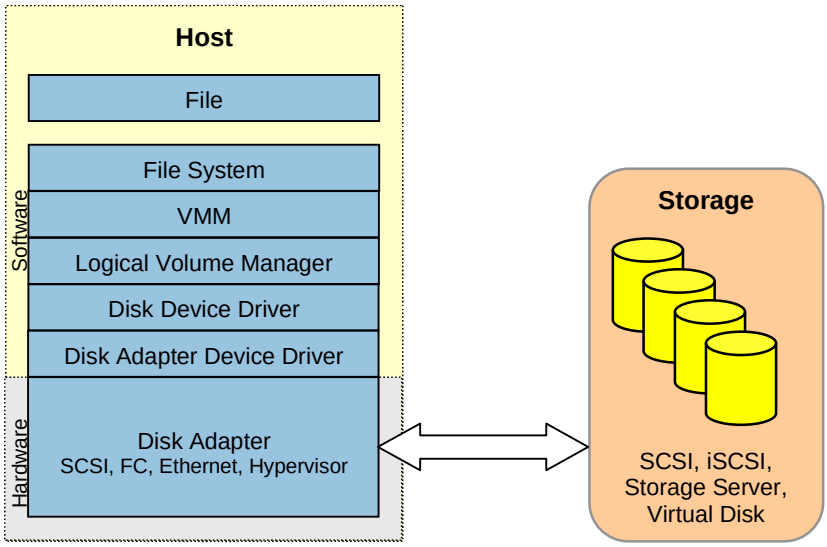


Figure 5.4.1 Storage Hierarchy

File

A file is a logically continuous stream of data. Applications can read from files and write into files without knowing anything about where the data on the file system is located at or what the block size is used by the file system to store the data. Each file has attributes like the filename, ownership, creating date, last modification data, file access permission, and others.

File System

The file system divides the available file system space into blocks of data; these blocks are known as the file system block size. File system data blocks have no structure. They are managed by inodes which associate file system data blocks with files. The inodes also contains the file attributes.

Logical Volume Manager

The Logical Volume Manager composes volume groups out of a single or multiple physical volumes. Each physical volume is divided into physical partitions which are mapped to logical partitions. Logical partitions are combined to logical volumes which can be used for file systems or raw data.

Disk Device Driver

The disk device driver adds the support for the protocol used by the storage devices. The most common storage protocol is the small computer system interface (SCSI) protocol. The disk device driver also controls the order in which requests are being serviced and when possible, coalesces multiple requests

Disk Adapter Device Driver & Disk Adapter

The disk adapter device driver adds support for the disk adapter that connects the computer to the storage device. Common disk adapters are SCSI, FC, SATA, and iSCSI adapters.

Storage

The storage can range from a single internal SCSI or SATA disk to a fiber channel (FC) attached storage server or Ethernet connected NFS/NAS server.

5.4.2 I/O Path

Figure 5.4.2 demonstrates the different I/O paths applications can use when accessing data located on a local storage device

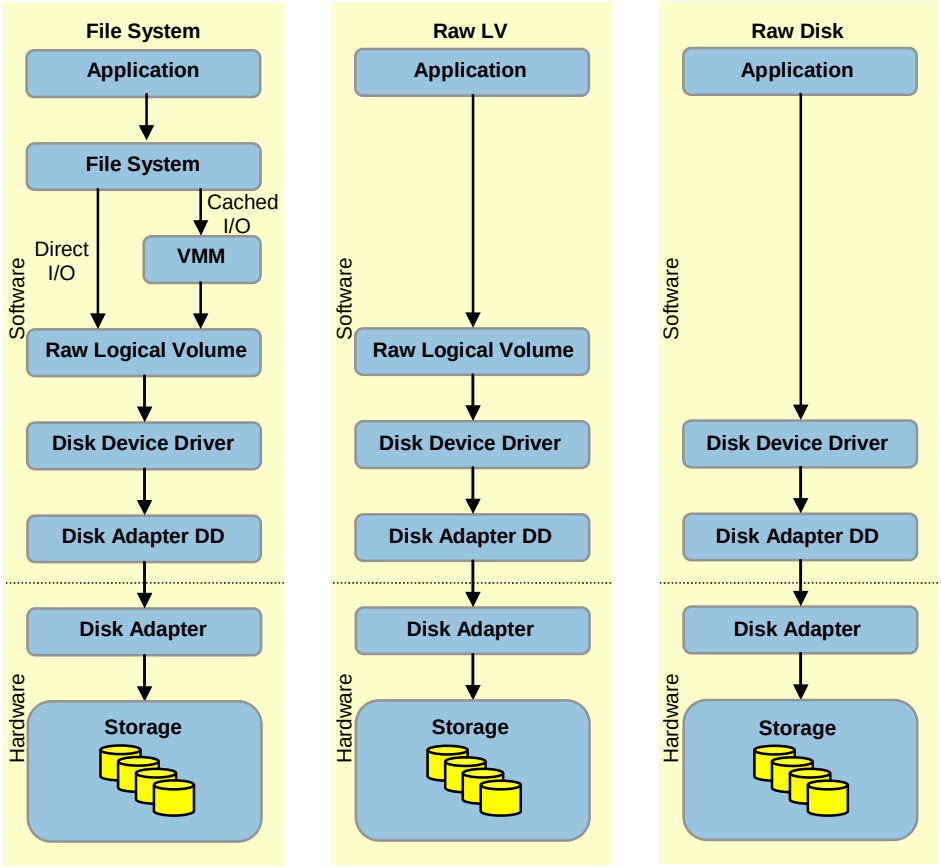


Figure 5.4.2 I/O Path

File System I/O

The most commonly used I/O path is file system I/O where applications read or write data that is managed by the file system. Applications can specify through the open flags whether the data of a file should be cached in VMM (default) or directly accessed, bypassing VMM.

The following table includes the most commonly used file access modes:

File Access Mode	Description
Non-synchronous I/O	Regular cached I/O (default unless specified otherwise); data is flushed out to disk through write behind and/or syncd; the file system reads pages into memory ahead of time when sequential read access pattern is determined
Synchronous I/O	Cached I/O, writes to files don't return until the data has been written to disk; the file system reads pages into memory ahead of time when sequential read access pattern is determined
Direct I/O	Bypasses the VMM file cache; data is read or written directly from the file system
Concurrent I/O	Same as Direct I/O but without inode lock serialization
Asynchronous I/O	I/O is serviced asynchronously by AIX kernel subsystem

Note: Direct I/O and Concurrent I/O file access mode also can be specified as mount option for the file systems.

Raw LV

Some applications, typically database applications, bypass the file system and VMM layers and access the logical volumes directly. Bypassing the file system and VMM layers usually is done to improve performance by reducing the path length.

Raw Disk

Applications can bypass LVM altogether by accessing the raw disks directly. Like with raw logical volumes, this typically is done to improve performance by reducing path length.

5.4.3 Network File System I/O Path

Similar to the I/O path for local storage, the data of a Network File System (NFS) can be cached by VMM (default) or accessed without caching by using the direct I/O mount option. The concurrent I/O option can also be used, which results in access similar to direct I/O, but without rnode lock serialization. Any operation on a Network File System is handled by the NFS client who communicates with the NFS server using the UDP or TCP network protocol. Please see chapter 6.5 Network Performance for details on the network layer and protocols.

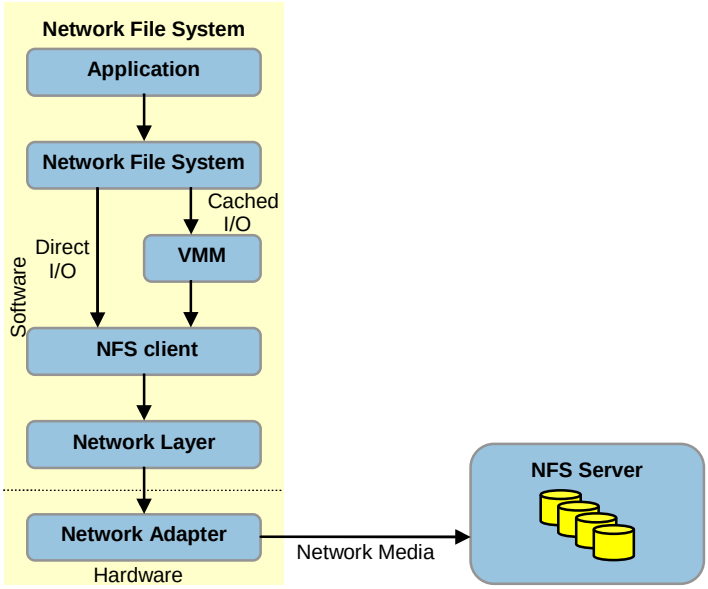


Figure 5.4.3 Network File System I/O Path

5.4.4 Storage Virtualization

The VIOS provides multiple options to virtualize storage to a vSCSI client, which are illustrated in figure 5.4.4.

Physical Volume Backed Virtual Storage

Physical volume backed virtual storage grants a vSCSI client access to a physical volume owned by a VIOS. This type of storage virtualization is common, and primarily used to export internally attached DASD or storage connected to a Fibre-channel adapter that does not support NPIV (N-Port ID Virtualization)

Raw Logical Volume Backed Virtual Storage

The VIOS allows physical volumes to be divided into raw logical volumes so long as the logical volumes within the VIOS are not striped or mirrored. This feature is typically used by customers that wish to take internal DASD, and divide it into logical volumes for vSCSI client partitions to use as boot devices.

File Backed Virtual Storage

Files within a J2 filesystem can be exported to a vSCSI client partition. This feature is not typically used by customers other than a method to provide access to vSCSI client partitions to optical media owned by a VIOS. The VIOS does not support buffered I/O, so none of the J2 files exported as virtual devices are stored within memory file cache.

N-Port ID Virtualization (NPIV)

NPIV provides the capability of creating multiple virtual instantiations of a single physical fibre-channel port. The VIOS owns the physical adapter and provides the pass through capability to the vSCSI client partition with the virtual fibre-channel adapter instance. A fibre-channel adapter and FC switch that support NPIV are required for this functionality. NPIV is very popular with customers, since it reduces the amount of administration required within the VIOS when changing the configuration of a vSCSI client, such as changing FC drive attributes.

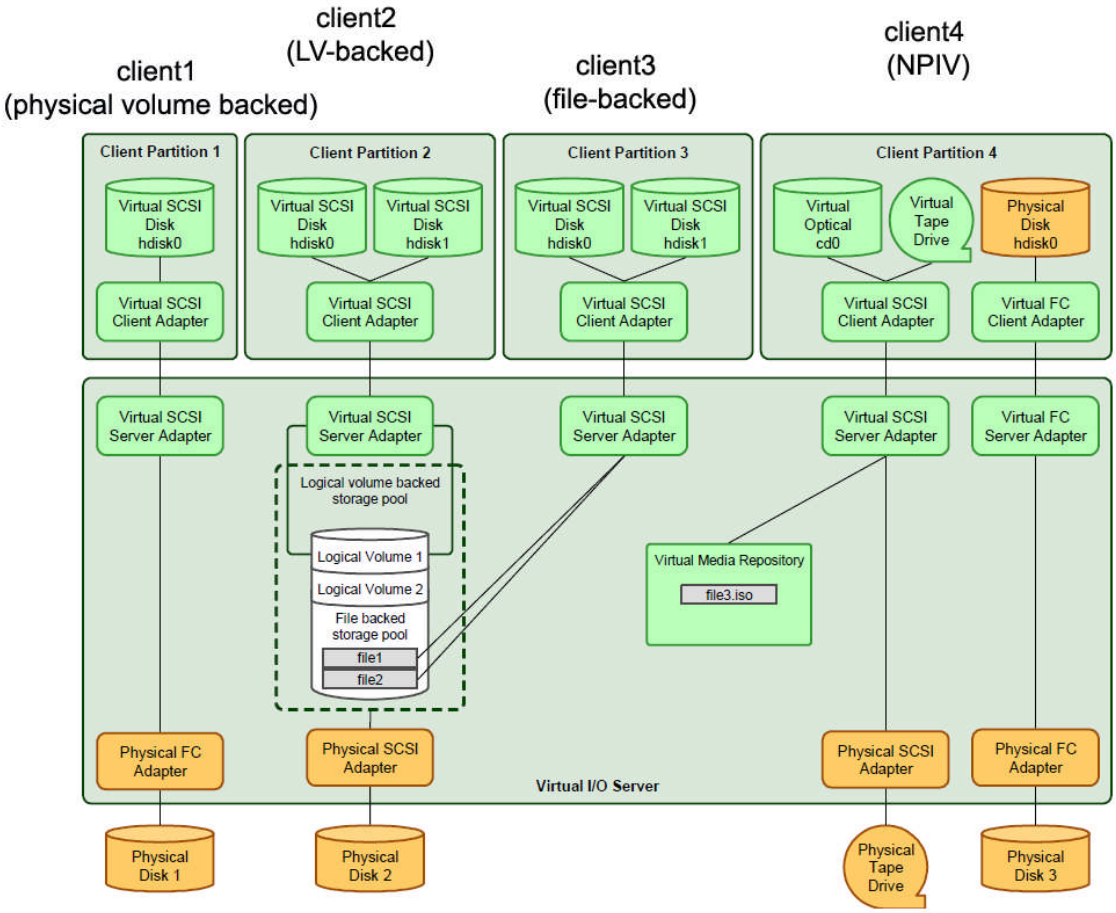


Figure 5.4.4 Virtual Disk I/O Options

5.5 Network Performance

5.5.1 Network Hierarchy

The following figure demonstrates the traditional network hierarchy for computer systems that are using the TCP/IP network protocols as communication vehicle.

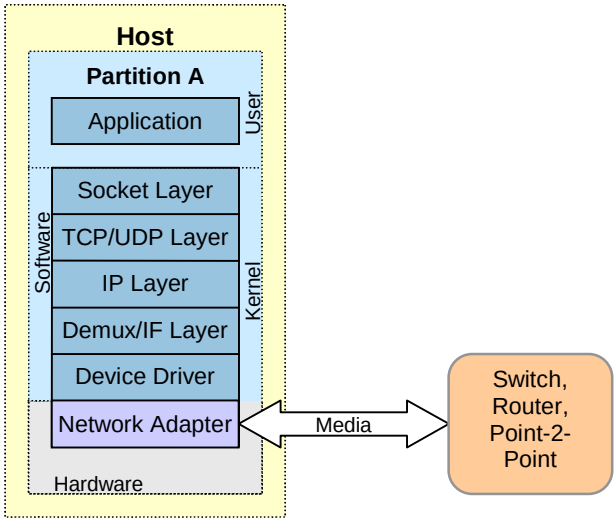


Figure 5.5.1 Network Hierarchy

Application and Socket Layer

Most applications that are communicating across networks are using the sockets API as the communication channel. A socket is an end point of a two-way communication channel. When an application creates a socket it specifies the address family, the socket type and the protocol. A new socket will be “unnamed” which means that it does not have any association to a local or remote address. In this state the socket cannot be used for a two-way communication.

In order to establish a two-way communication the application first needs to use the bind() subroutine to specify the local address and port or the local path name. The second step depends on whether the application is connecting to or waiting for connection requests from remote systems. In the case of connecting to a remote system the application needs to call the connect() subroutine which adds the address and port number of the remote system to the socket and also tries to establish a connection with the remote system if a connection oriented protocol is being used. To accept connection requests from remote systems the application need to call the accept() subroutine.

TCP/UDP

The Transmission Control Protocol (TCP) is a bidirectional and connection oriented protocol. It provides a reliable, sequenced and unduplicated flow of data. Applications that are using TCP must establish a connection to the remote system.

The User Datagram Protocol (UDP) is a bidirectional and connectionless protocol which does not require establishing a connection between two endpoints prior communication. It does not provide flow control and therefore does not detect packet loss, packets that are out of order or duplicate packets.

IP Layer

The Internet Protocol (IP) layer provides a basic datagram service to higher level protocols such as TCP and UDP. When a higher level protocol gives a packet to the IP layer that is larger than the Maximum Transfer Unit (MTU) of the network interface the packet will be split into multiple fragments and send to the receiving machine. The receiving machine will reassembles the fragments into the original packet size before sending it to the higher level protocols.

Demux / Interface Layer

The Interface layer receives outgoing packets from the IP layer. It places the packets on a transmit queue which is processed by the interface device driver.

The Demux layer receives incoming packets from the network interface device driver and calls the IP layer for input processing. By default, the IP input processing is done on the interrupt thread but can be queued for off level processing by using dog threads. When dog threads are used, all IP, TCP and socket code processing for incoming packets will be done by the dog threads.

Network Adapters

High speed network adapters, like 1/10 Gigabit Ethernet adapter, support performance features like large send and checksum offload to offload processing cycles from the CPU to the network adapter.

Network Media

The most commonly used network media today is the Ethernet at various speeds including 10 megabit, 100 megabit, 1 gigabit and up to 10 gigabit. Other media types include InfiniBand, Token Ring, ATM, X.25, FDDI and Serial Line.

5.5.2 Network Virtualization

In a virtualized environment, physical network adapters are replaced by virtual adapters that communicate with other system through the Hypervisor instead of a physical media. Figure 5.5.2 demonstrates the network hierarchy in a virtualized environment.

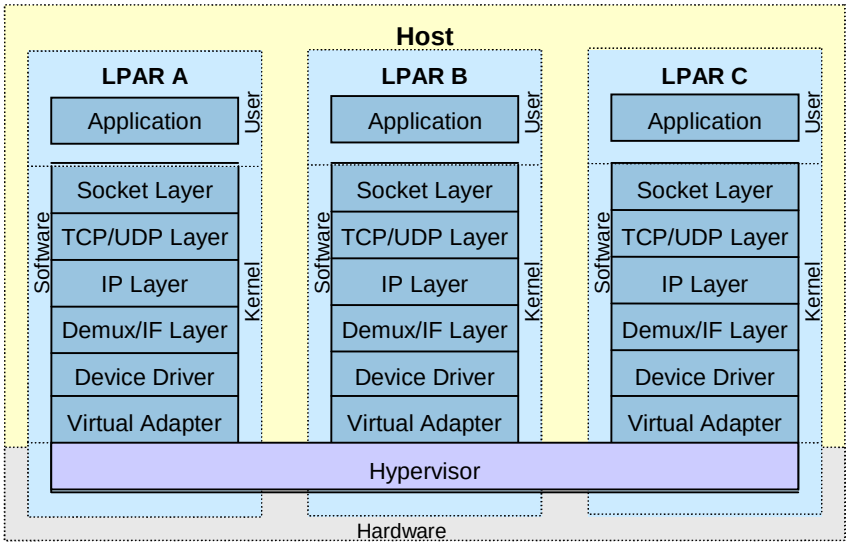


Figure 5.5.2 Network Virtualization

5.5.3 Shared Ethernet Adapter

A Shared Ethernet Adapter (SEA) allows LPARs to communicate with an external network by bridging the internal VLAN to the VLANs on the physical network. The SEA allows bridging one or more virtual adapters to a physical Ethernet Adapter, a Link Aggregation or Etherchannel device.

Figure 5.5.3 demonstrates the SEA bridging the VLAN for LPAR A and LPAR B to a physical Ethernet.

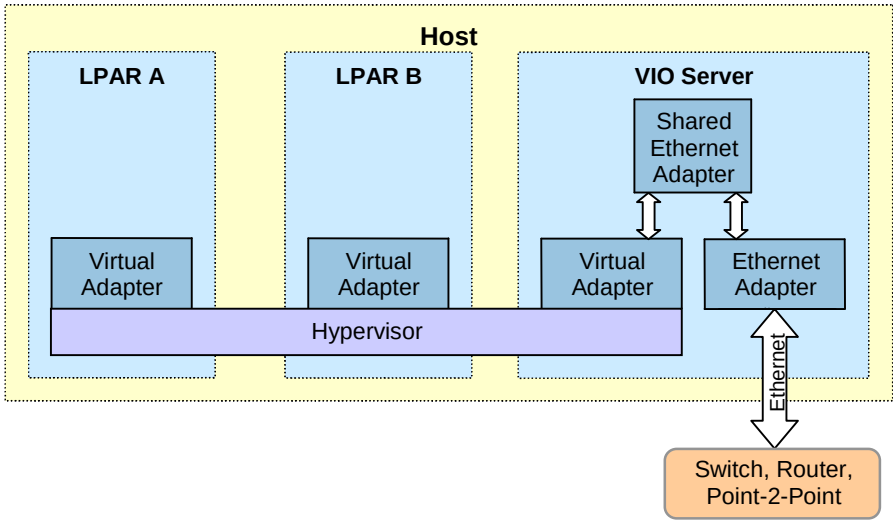


Figure 5.5.3 Shared Ethernet Adapter

5.5.4 Host Ethernet Adapter

POWER6 based machines provide the Host Ethernet Adapter (HEA) feature, also known as Integrated Virtual Ethernet adapter (IVE adapter), which allows sharing of physical Ethernet

adapters across multiple logical partitions. A Host Ethernet Adapter is connected directly to the GX+ bus and offers high throughput and low latency.

Logical partitions connect directly to Host Ethernet Adapters and can access external networks through the Host Ethernet Adapter without going through a Shared Ethernet Adapter (SEA) or another logical partition.

Figure 5.5.3 demonstrates the network hierarchy for logical partitions that communicate to an external network through a Host Ethernet Adapter

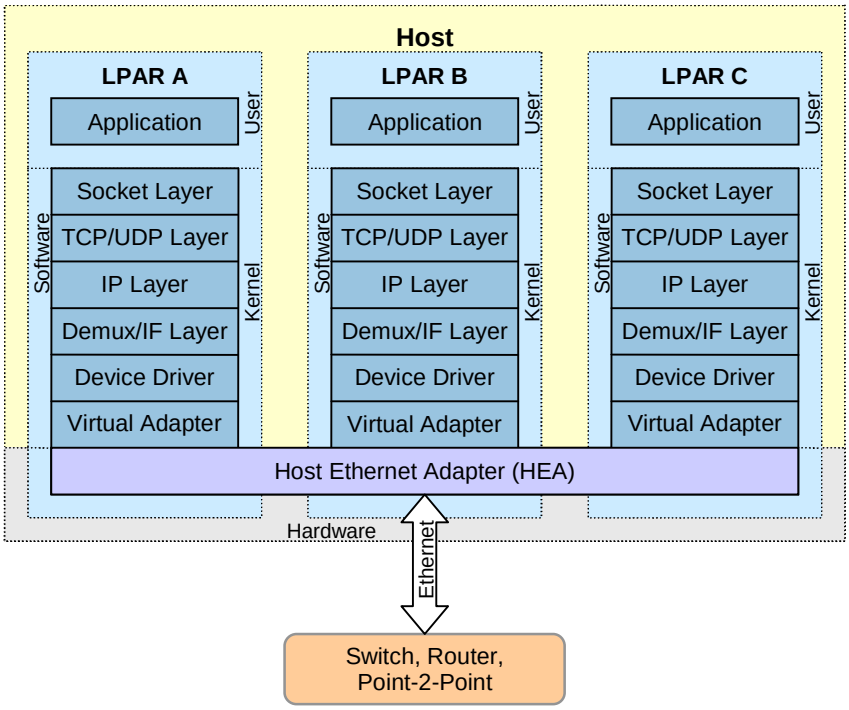


Figure 5.5.4 Host Ethernet Adapter

5.6 Software Performance

Many performance issues can be traced back to application development approaches that deviate from best practice. While a detailed discussion of such best practices lies outside the scope of this document, generally speaking, application development best practices can be grouped into the following categories: employ up-to-date software; when compiling C/C++/Fortran applications, use at least -O optimization; employ Power-specific and AIX-specific optimizations; and adopt a rigorous testing discipline. Each of these is treated in additional detail below.

5.6.1 Employ up-to-date software

In general, recent versions of IBM software better exploit the features of more recent Power systems, and thus it is advantageous to employ the most up-to-date IBM software whenever

possible. The converse of this is that older versions of IBM software generally do not exploit the advanced features of recent Power systems, and when those older versions are used, a less-than-optimal customer performance experience often results. An example of this is AIX Version 5.3, which only supports two SMT threads per core. When AIX 5.3 is run on a Power7 system, two threads per core are disabled, and performance may be impacted as a result. A complete list of the most recent versions of the spectrum of software products is beyond listing here, but here are some general guidelines:

- Applications should always be developed on AIX 5.3 or newer versions of AIX. Deployment of applications on Power7 should be on the latest technology level/service pack of AIX V6 or V7, and should never be on an AIX 6.1 technology level older than TL5.
- When using the IBM XL C/C++ compilers, V10 or V11 should be used. Power7 exploitation was introduced in V11. V9 of the compiler is scheduled to be withdrawn from support on September 30, 2012. V8 or older versions of the compiler have been withdrawn from support and should not be used.
- Versions of Java including JDK6 SR7, or newer, should be used, as those versions exploit Power7. Power7 exploitation was introduced into WebSphere Application Server with V7; however, additional optimizations in V8 make that the preferred choice. WebSphere Application Server V6 or older versions do not contain Power7 exploitations and should be considered candidates for upgrade on Power7 deployments.

5.6.2 Compiler Optimization

When compiling C/C++ applications, use at least -O optimization. The XL compilers provide a plethora of options to control the degree of optimization of the generated machine code. However, one simple rule is sufficient to provide a reasonable level of optimization for most applications: Use at least the -O optimization flag. Following this rule should be a paramount consideration for application developers, as the difference between not using any optimization flag, and using the -O flag, is large -- up to a 10x performance increase in application performance can be observed when -O optimization is employed.

5.6.3 Platform specific optimization

Employ Power-specific and AIX-specific optimizations. The Power architecture and AIX include specific optimizations that are not available on other platforms or operating systems. One Power-specific feature that frequently yields a performance increase is the utilization of 64KB hardware pages. In the category of AIX-specific optimizations, multi-threaded applications (i.e. applications where a process includes many software threads) and that utilize the 'malloc' memory allocator should evaluate using the multi-heap malloc option.

5.6.4 Software testing

Employ a rigorous testing methodology utilizing the most current technology. Whereas exploitation of specific Power and AIX features usually has a beneficial impact on performance, it's also the case that advancements in Power technology can expose latent application-level performance issues that did not surface on previous generations of technology. For example, the doubling of the number of SMT threads on Power7 (from two on Power6 to four on Power7)

doubles the number of application threads that can run simultaneously. While this usually provides an improvement in performance, it can also allow the application to run at a higher degree of "scale" than was previously tested, and this, in turn, may expose scalability issues in the application. For this reason, it is advisable to do performance and scalability testing on the latest available Power technology, and on systems that are of the scale than can cover customer deployments.

5.7 Performance Metrics

5.7.1 System Components Performance Metrics

In addition to the metrics reported by performance benchmarks, the performance of computer systems is mainly measured through the use of analysis tools. The following represents a high level overview of the main performance metrics for various system components:

- CPU
  - %user, %system, %idle, and %wait
  - Physical Consumed, Entitlement
  - Number of context switches, interrupts, and system calls
  - Length of the run queue
  - Lock contention
- Memory
  - Virtual memory paging statistics
    - Paging rate of computational pages
    - Paging rate of file pages
    - Page replacement page scanning and freeing statistics
    - Address translation faults
  - Cache miss rates
- Disk I/O
  - Amount of data read/written per second (KB, MB, or GB per second)
  - Transactions per second
  - Elapsed time for I/O to complete
  - Queuing time
- Network I/O
  - Amount of data transmitted/received per second (KB or MB per second)
  - Number of packets transmitted/received per second
  - Network CPU utilization
  - Network memory utilization
  - Network statistics, errors, retransmissions

6 Performance Analysis and Tuning Process

6.1 Introduction

This chapter covers performance analysis and tuning process from a high level point of view. Its purpose is to provide a guideline and best practice on how to address performance problems using a top down approach.

6.1.1 Performance monitoring before a problem occurs

Application performance should be recorded using log files, batch run times or other objective measurements. General system performance should be recorded, and should include as many components of the environment as possible.

Performance statistics should be collected based on the typical period of activity in your environment. The minimum period should be considered one week in an interactive system, as typical performance may vary based upon the day of the week. A batch processing system that typically runs large end of month reports should consider a month as the minimum period. Data should be collected and stored over multiple periods to allow for comparison.

Performance monitoring of the environment provides many benefits. The application performance will provide a useful method of quantifying the performance change. A history of acceptable system performance will focus analysis on changed components. Also, continual review of performance monitoring will show trends in performance and capacity before they become system wide problems.

6.2 Defining a Performance Problem (What is slow?)

Before collecting any data or making tuning or configuration changes, define what exactly is slow. A clear definition about what aspect is slow usually helps to shorten the amount of time it takes to resolve a performance problem since a performance analyst gets a better understanding what data to collect and what to look for in the data.

A good way to define a performance problem is to answer questions like:

- Is everything slow or just a specific task?
- What application log file demonstrates the performance problem?
- Is it only slow when run remotely but fast running locally?
- Is it slow for everyone or just for a single user?
- and many more questions

Please refer to section “Questions that help IBM diagnose the problem” in chapter 9 of this document for a list of questions asked to define a performance problem.

6.3 Top-down Performance Analysis

After defining the performance problem, the next step is to collect performance data to start with the analysis. There are many performance data collection tools available that are useful to determine the cause of a performance problem. However, there is not a single tool available that would cover all aspects of computer performance.

For example, a typical system performance monitoring tool provides information about CPU utilization, memory usage, disk utilization and network throughput; however it usually does not provide detailed application specific information.

For most cases, application performance issues are most efficiently being addressed by using a top-down approach. The performance analysis using a top-down approach starts at the place the performance problem surfaced, which typically is the application, and walks its way down through the lower layers towards the hardware.

Top-down approach layers:

1. Application including middleware and libraries
2. System
3. System Components and Kernel

6.3.1 Application

Many applications and middleware provide logging or tracing facilities that are useful to determine if a performance problem is caused by something within the application, the operating system or both.

For example, some databases provide performance reports with information about the overall efficiency, such as database cache hit ratio, most frequent events within the database or even detailed information about individual queries. This information is useful to determine whether a database query was running slow due to an issue within the database itself or due to system resource bottleneck.

If available, an application logging or tracing facility should be utilized first to determine if the probable cause of the performance problem lies within the application or if further analysis of the system is required.

For applications without logging or tracing facilities, system performance analysis can help to determine the cause of an application performance problem. However, it should be understood that system performance analysis without any information about what system resource, if any, an application is waiting on, mainly is an approach to determine system resource bottlenecks.

When analysis of the application is not performed, further analysis of system performance may reduce system resource bottlenecks. However, this is not optimal and will not directly address all bottlenecks with the application.

6.3.2 System

The goal of a system performance analysis is to identify resource bottlenecks that have a negative impact on system and/or application performance. If a problem is well defined, for example ‘reads from the FC connected storage server take a long time to complete”, the initial analysis can concentrate on those system resources that can have a direct impact.

System performance analysis is done through monitoring system resources and performance counters that are maintained by the operating system. AIX provides a variety of performance monitoring tools that are useful to determine system resource bottlenecks. Typical system resources to look at are CPU, memory, disk and network.

When a system resource bottleneck has been identified performance tuning or adding resources to the system might solve the problem. A good example of adding resources to solve a performance problem is to increase the amount of real memory when paging to paging space has been identified to be caused by ‘over committed’ memory, i.e., the amount of active virtual memory exceeds the amount of real memory.

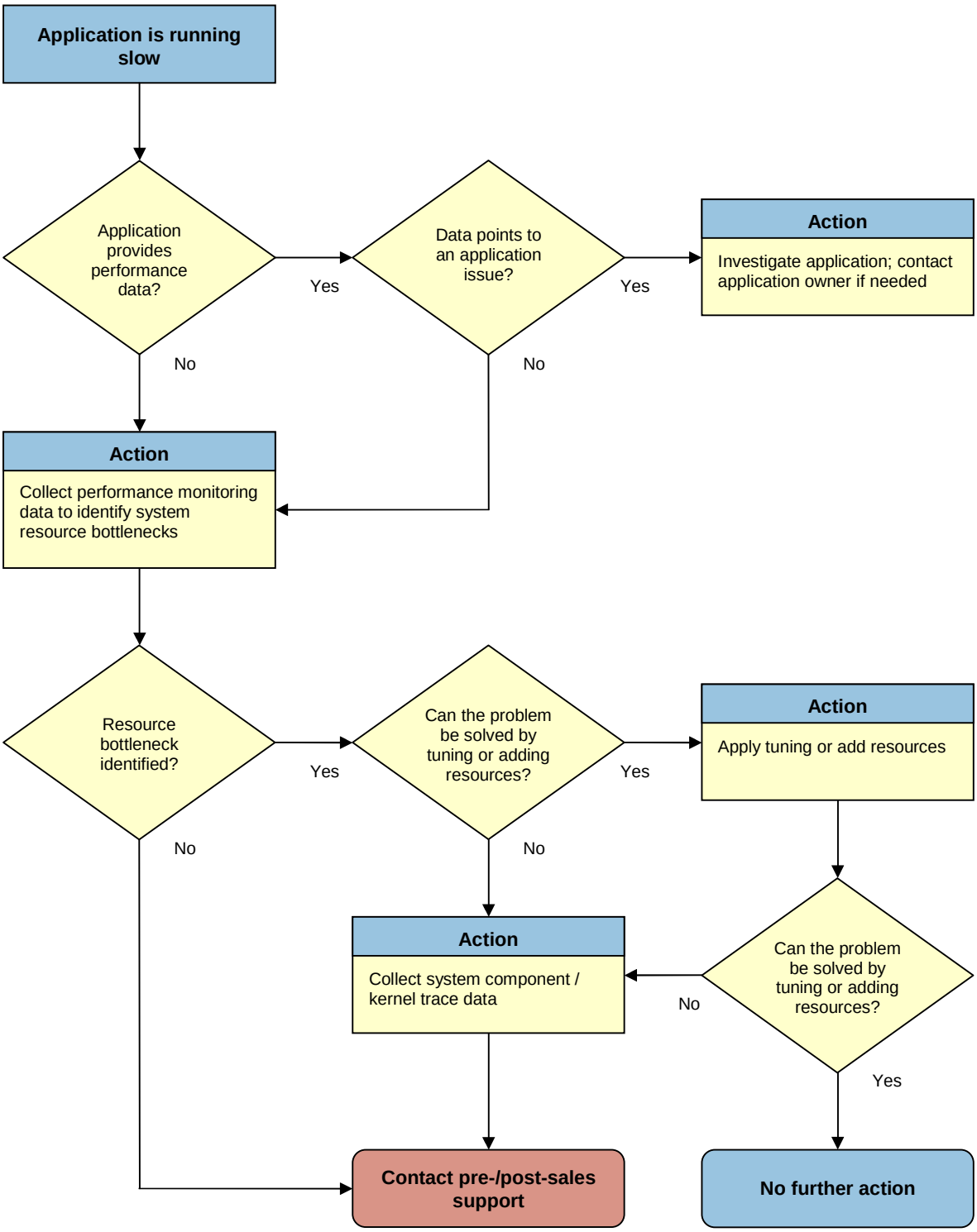
6.3.3 System Components and Kernel

While performance monitoring tools work well to determine performance bottlenecks they often only show the symptom but not the root cause of a performance problem. Performance problems can occur where adding resources or changing tunable settings does not solve the problem and in some cases makes it even worse.

For example, the system shows a high CPU utilization due to lock contention. Adding more CPUs can further slow down the system because each additional CPU might compete for the same lock.

When a performance problem cannot be solved by tuning or adding resources, or when the root cause needs to be identified prior to any change, system components and kernel needs to be analyzed. AIX provides a trace facility that allows tracing of selected system events such as kernel routines, kernel extensions, interrupt handlers, etc.

6.3.4 Top down performance analysis flow chart



7 Performance Analysis How-To

7.1 Introduction

This chapter is intended to provide information and guidelines on how to address common performance problems seen in the field, as well as tuning recommendations for certain areas.

Please note that this chapter is not intended to explain the usage of commands or to explain how to interpret their output.

- How to tune VMM page replacement to reduce paging
- How to address CPU bottlenecks using tprof
- How to address paging issue
- How to address NFS sequential read/write performance problems
- How to migrate from cached database environment to concurrent I/O
- How to tune TCP/IP for Superpacket IB interface

7.2 How to tune VMM page replacement to reduce paging

Paging is one of the most common reasons for performance problems. Paging means that computational pages are ‘stolen’ by the VMM page replacement algorithm to free memory and written out to the system paging space.

The following VMM page replacement tunable settings will allow the system to use up to 90% of its real memory for file caching but favors computational pages over file pages. Page replacement may steal computational pages once the percentage of computational pages in memory exceeds 97%; i.e. 100% - minperm%.

AIX 5.3 defaults	AIX 5.3 recommended values and AIX 6.1/7.1 defaults
`minperm% = 20`	`minperm% = 3`
`maxperm% = 80`	`maxperm% = 90`
`maxclient% = 80`	`maxclient% = 90`
`strict_maxperm = 0`	`strict_maxperm = 0`
`strict_maxclient = 1`	`strict_maxclient = 1`
`lru_file_repage = 1`	`lru_file_repage = 0`
`page_steal_method = 0`	`page_steal_method = 1*`

* Changing the page_steal_method is an optional tuning for systems that are utilizing only a small percentage of their memory for file caching.

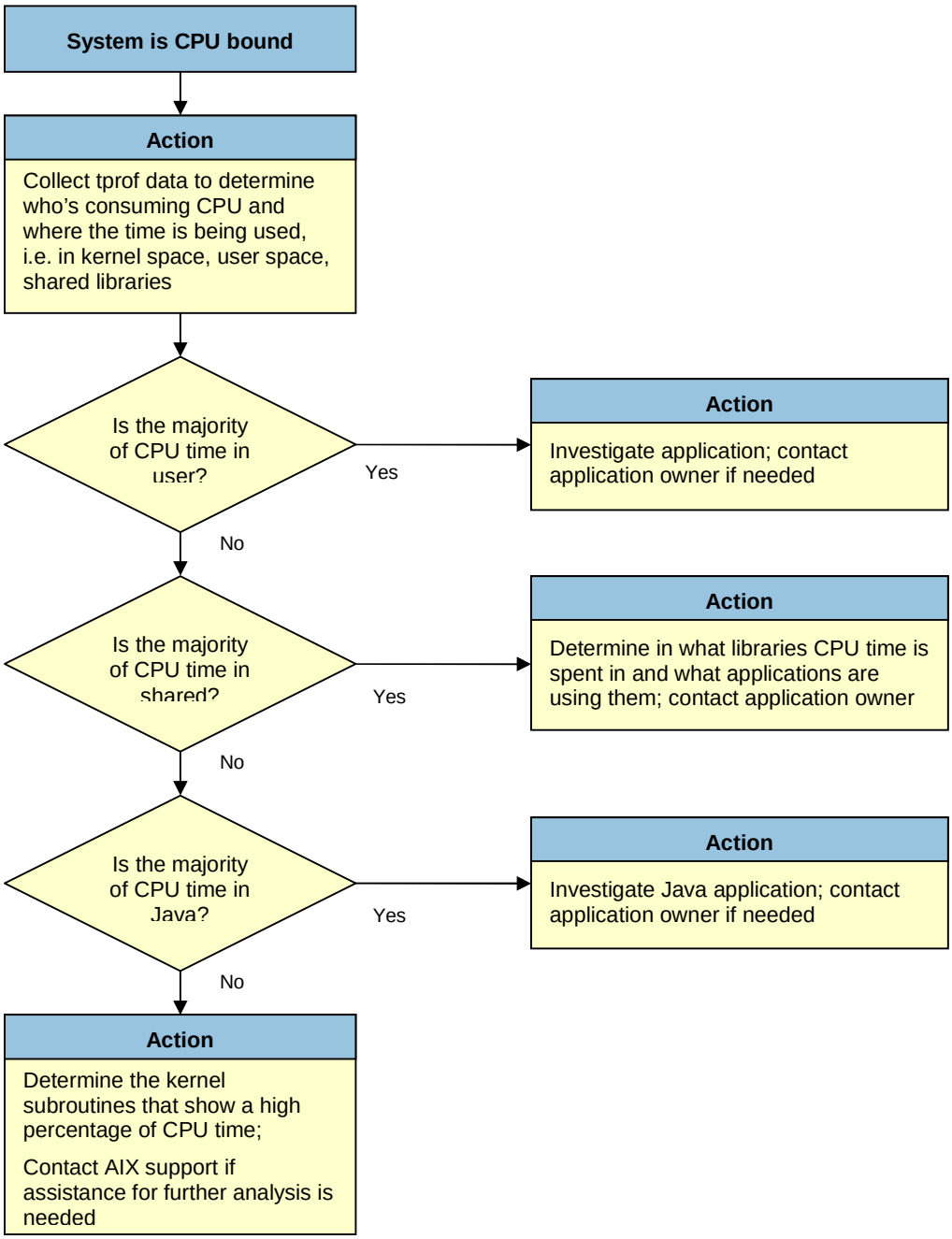
Note: the values for the above listed tunables became the defaults as of AIX 6.1.

Refer to chapter 8.5 on How to address paging issues.

7.3 How to address CPU bottlenecks using tprof

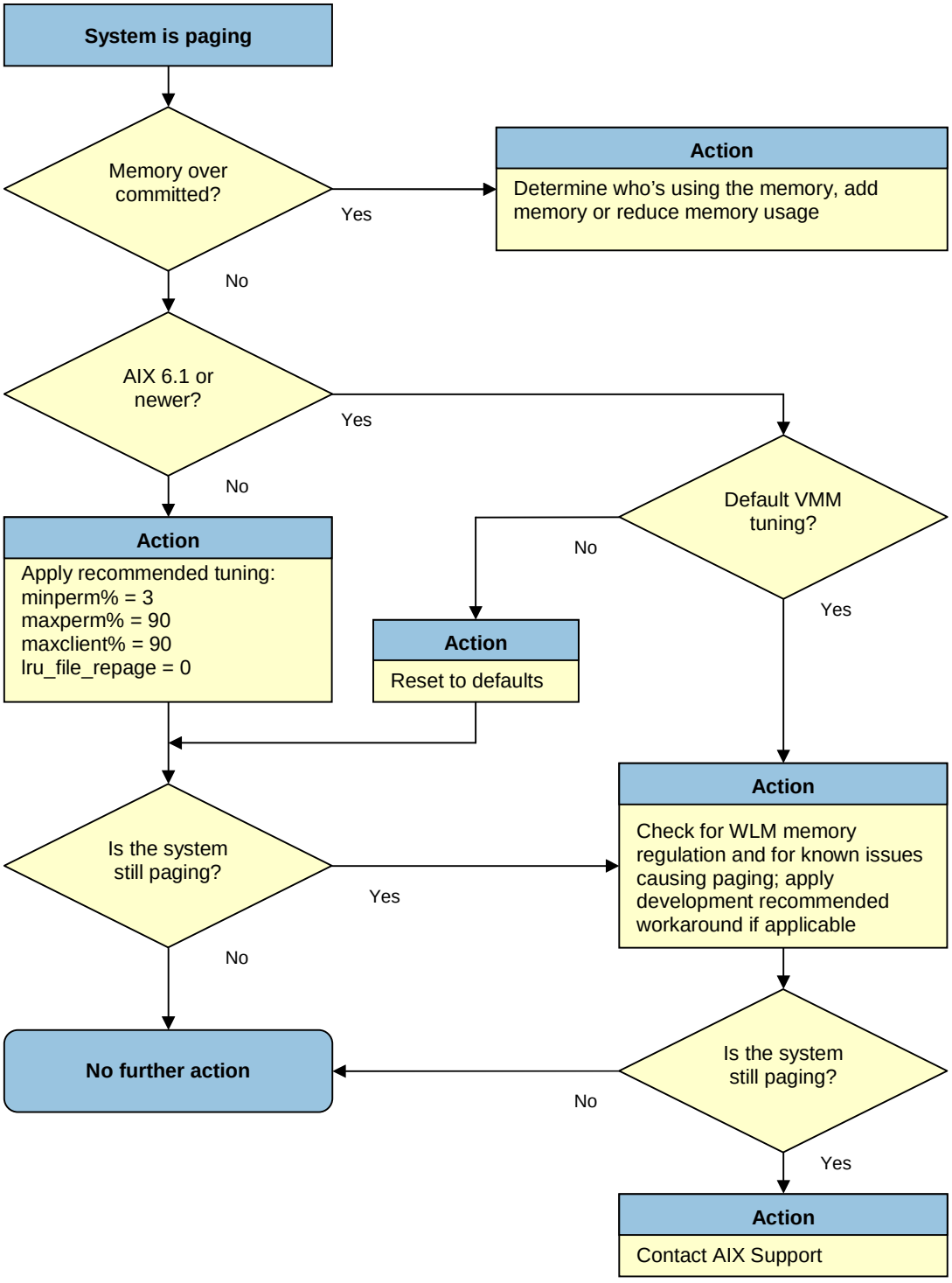
While many performance monitoring commands report the CPU utilization as user, kernel, idle, and wait time, they usually do not provide much information on who is consuming CPU time and where the CPU time is spent in. The kernel trace based tprof command can be used to get this level of detail.

The following flow chart is a guideline on what actions to take based on the CPU usage information reported from the tprof command:



7.4 How to address paging issues

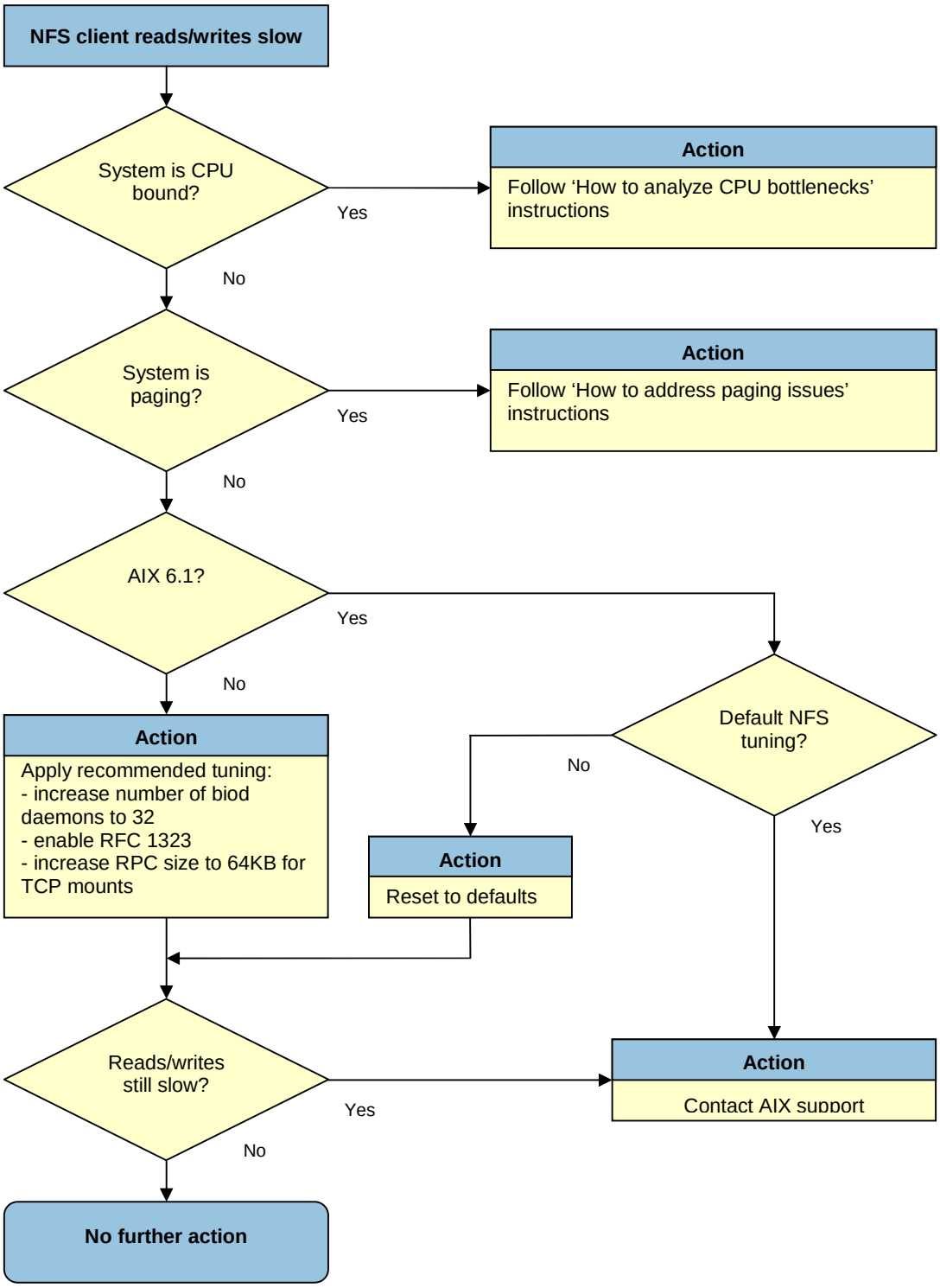
The following flow chart is intended to provide a high level best practice guideline on how to address paging issues on AIX 5.3, 6.1 and 7.1.



7.4.1 What's causing paging?

- System memory is over committed, i.e.,
  - the amount of active virtual memory (avm in vmstat, virtual in svmon -G) exceeds the amount of real memory
  - the percentage active virtual memory exceeds 100% - minperm%
- The percentage of memory used for file caching is between the minimum and maximum thresholds and LRU uses repage ratio for computational pages and file pages to decide what pages to steal:
  - JFS: numperm is between minperm% and maxperm%
  - JFS2, NFS: numclient is between minperm% and maxclient%
- WLM classes reach their hard memory limit
- Files opened with the deferred update flag O_DEFER will be written to paging space instead of being written back to the file system.
- List-based LRU will steal computational pages when it detects memory starvation
- Legacy: system is running out of free pages for an extended period of time on system with JFS file systems

7.5 How to address NFS sequential read/write performance problems



7.5.1 What is causing slow NFS sequential read/write performance?

It is important to keep in mind that there are multiple hardware and software components between an application on an NFS client seeing “slow” NFS sequential read/write performance and the storage subsystem on the NFS server where the data being read/written resides. What may seem like a problem caused by poor NFS tuning might instead be due to a configuration or tuning issue at: (1) the TCP/IP layer, (2) physical network layer, (3) the storage subsystem connected to the NFS server. Therefore, some problems to look for to identify those components as the potential cause, or to rule them out, include: (1) dropped packets on the network, (2) errors on the storage subsystem or SAN.

7.6 How to migrate from cached file system to concurrent I/O

For many database workloads, using concurrent I/O instead of cached file system results in better performance due to a shorter path length and the ability for the database application to make multiple updates to the database files concurrently.

However, a common pitfall when moving from cached file system to concurrent I/O in a database environment is that the impact of the cache hit ratio of the database cache together with the file system cache hit ratio is often underestimated. Database jobs that benefited from finding data in the database cache and the file system cache might run slower after moving to concurrent I/O due to the missing file system cache.

The purpose of this section is to explain the reason for getting worse performance when moving from cached file system to concurrent I/O and provide a guideline on how to prevent to run into this pitfall.

7.6.1 Effective Cache Hit Ratio

Databases running on cached file system are effectively using two caches. One cache is the database internal data cache and the other is the file system cache which contributes to the overall, or effective, cache hit ratio.

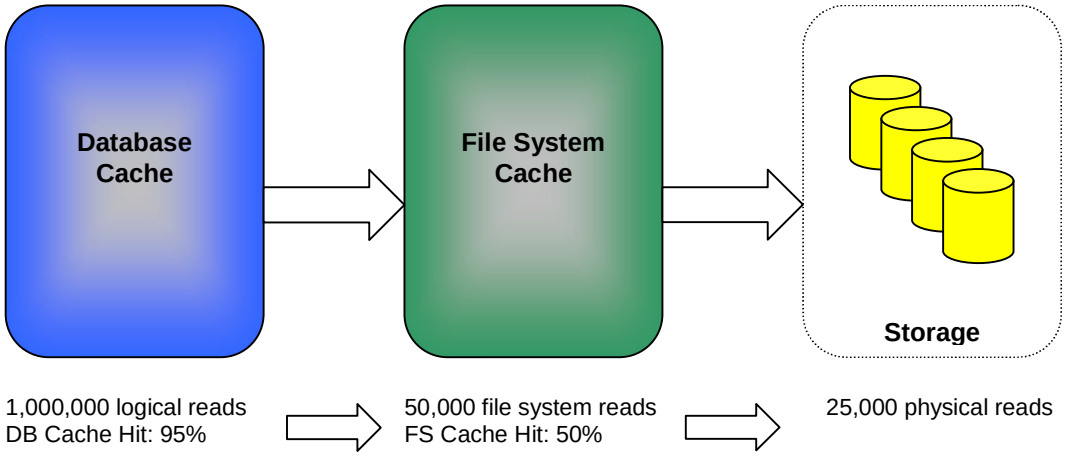
The cache hit ratio of a database is the percentage of logical reads that are satisfied by data that already in the database cache. A higher cache hit ratio usually results in better performance (there are exception to this, which are not covered by this section). The percentage of logical reads that are not satisfied with data in the cache has to be read from the operation system. A database typically reports those reads as physical reads.

Physical reads from a database using cached file system may or may not result in physical reads from the storage device. No physical I/O will occur if the read can be satisfied with data that is already in the file system cache. Data that is not cached or partially cached will result in a physical read. The more data is cached in the file system cache the fewer physical read will occur and as a result, the cache hit ratio of the database cache become less important since the data is in memory anyway.

Physical reads from a database using raw logical volumes, raw disks or concurrent I/O always result in physical reads from the storage device since none of the data is cached in memory; therefore the database cache hit ratio becomes more important.

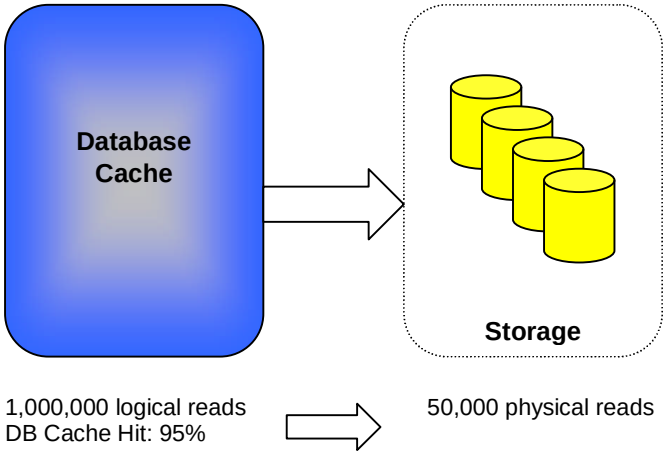
The following figure represents an example of the effective cache hit ratio for a database environment using cached file system.

In this simplified example, the database reports 1,000,000 logical reads and a DB cache hit ratio of 95%; about 50,000 reads are physical reads from the file system. Out of the 50,000 “physical” reads from the database, 50% are satisfied from data already in the file cache and don’t require any physical reads from the storage device. The remaining 50% that are not cached result in 25,000 physical reads from the storage device. Thus the file system cache contributes with 2.5% to the effective cache hit ratio which is 97.5% in this example.



With an average read response time of 10 milliseconds from the storage device, the database would report 5 milliseconds the average response time for physical reads since half the number of physical reads is satisfied from the file system cache.

The following figure shows the impact on the effective cache hit ratio when moving from cached file system to concurrent I/O.



Effective Cache Hit: 95%

The effective cache hit ratio now is equal to the DB cache hit ratio since all physical reads from the database result in physical reads from the storage device.

The database now reports an average physical read response time of 10 milliseconds instead of 5 milliseconds which is misleading since the physical read response time from the storage device didn't change.

7.6.2 How to determine the effective cache hit ratio

The most accurate method to determine the percentage of database physical reads from the file systems is satisfied from data already in the file system cache would be through tracing the kernel. However, this method is time intensive and requires deep knowledge in kernel trace performance analysis.

An easier method but less accurate is to compare the number of physical reads reported by the database with the number of physical reads on the storages devices on which the database data files are located.

For example, if the database reports 10,000 physical reads and the storage devices shows 5,000 physical reads.

As a rule of thumb, the closer the average physical read size on the storage devices is to the database record size, the more accurate the result.

7.6.3 How to avoid the pitfall

The way to avoid this pitfall is to increase the amount of memory available to the database for caching to the point that the database cache hit ratio is equal (or better) to the effective cache hit ratio of database and file system cache.

The AIX filemon command can be useful to estimate the amount of additional memory that is needed by determining how much of the file cache is used for database files. When using filemon it is important that the data is collected during typical workload activity.

7.7 How to tune TCP/IP for Superpacket IB interface

With the release of AIX 5.3 TL8 and AIX 6.1 TL1 versions of AIX we introduced a new AIX IP driver for Infiniband adapters. This driver improves the bandwidth performance of TCP/IP over both the Single Data Rate (SDR) and Double Data Rate (DDR) class adapters. Along with the Superpacket driver we extended the High Performance Switch global interface ml0 to include IB adapters. The global interface is similar to Etherchannel where is uses multiple links of IB through a single interface. Along with these improvements come new tuning guidelines to achieve peak performance.

7.7.1 Recommended feature usage based on machine type

With this release there is a new mode in the IP driver for Infiniband. This mode, called Superpacket mode, improves the bandwidth for large data transfers using IP. There are also additional performance gains for larger machines from turning on the threads setting for the interfaces. The following are recommendations on what settings to use depending on configuration of the machine or LPAR you are using.

Setting	Up to physical 8 CPUs	Greater than 8 physical CPUs
Superpacket	On	On
Threads	Off	On

To turn on Superpackets for each interface used, run the following command:

```
chdev -l ibx -a superpacket=on
```

Then do a mkdev to make the interface pick up the changes

```
mkdev –I ibx
```

where x in ibx is the interface number (ib0, ib1.....)

The Superpacket setting is preserved across reboots so only has to be done once.

To add threads to an interface use the following command:

```
ifconfig ibx thread
```

where x in ibx is the interface number (ib0, ib1.....)

The thread setting is not preserved across reboots so has to be run again after a reboot.

1. To get Superpacket performance all nodes being used have to be set for Superpacket mode.
2. If one node is Superpacket and the other one is not you will get IPoIB-UD (IP over Infiniband - Unreliable Datagram) performance which is much slower.
3. Superpacket mode is only available on AIX and will not be used communicating to non-AIX systems over IB.

7.7.2 Tuning for best MTU size of the IB interfaces

You should not have to set the MTU of the Infiniband interfaces on most systems. Changing the MTU value from the default settings can have unexpected consequences so it is not recommended except for the most advanced user.

When the Infiniband interface is initially configured, it queries the IB switch to get the MTU size to use. For all SDR adapters it will be 2048. For the DDR adapters, if the switch is set to 4K MTU and you specify the 4K MTU multicast group p_key on, you will get a MTU of 4096. If the switch is not set to 4K MTU or you did not specify the 4K MTU multicast group p_key on, you will get a MTU of 2048.

When you turn on superpackets (described earlier), the MTU of the Infiniband interfaces should automatically change to 65532. This will give the best bandwidth for most situations. If you change the MTU setting on the Infiniband interfaces, the Superpacket MTU setting will use this new MTU.

7.7.3 Configuring ml0 to match the performance tuning on IB interfaces

When using the ml0 global interface across multiple Infiniband interfaces your will need to manually set the network options to match the expected settings for the ibx interfaces. The no settings you need to set should be:

Non-Superpacket

```
rfc1323=1
tcp_sendspace=262144
tcp_recvspace=262144
```

Superpackets

```
rfc1323=1
tcp_sendspace=524288
tcp_recvspace=524288
```

These setting changes can be made through no or the nextboot tuning facility. Consult the AIX performance management tuning documentation for more information.

8 Frequently Asked Questions

8.1 Introduction

This chapter covers frequently asked questions from the field.

8.2 General Questions

8.2.1 What are the general performance tuning recommendations for best AIX performance?

There is no such thing as a general performance tuning recommendation.

8.2.2 I heard that... should I change...?

No, never apply any tuning changes based on information from unofficial channels. Changing performance tunables should be done based on performance analysis or sizing anticipation.

8.2.3 I found a three year old best practice paper on the web, is it still valid?

Probably not... Most of the best practice papers and recommendations are written for the versions of the applications and operating system that were current at the time the paper is written. Updates or newer versions of the applications and/or operating system often introduce changes and improvements that requires different, sometime less, tuning.

Please contact the author and/or the application and operating system to obtain the latest best practice recommendations.

8.2.4 Why don't I see all tunables on AIX 6.1?

AIX 6.1 introduces the concept of restricted tunables. Restricted tunables are tunables which should not be changed unless recommended by AIX development or AIX development support. The tuning commands, like vmo, do not show restricted tunables by default. To show all tunables add the -F flag to the tuning command.

Note: changing a restricted tunable permanently requires a confirmation and will be shown as an informational error log entry at system boot time.

8.2.5 Do I need to recompile my application to get good performance on a new platform?

For some applications, there may be gains from exploiting either more aggressive options in the existing compiler or moving to a more recent version of the compiler. For example, on the existing compiler, gains may result if a compilation is able to specify a narrower range of target processors (-qarch, -qtune), request higher optimization levels (e.g. -O3 vs. -O2), inlining some key routines, or refine aliasing information.

8.3 CPU Questions

8.3.1 What is I/O wait?

It's a common misunderstanding that I/O wait is the time a CPU is waiting for physical I/O to complete. The reality is that I/O wait is nothing more then CPU idle time during which physical

I/O to a local disk or remotely mounted disk (NFS) was outstanding and the CPU is still available for computational work. A high percentage of I/O wait time does not indicate a performance problem without further I/O statistics indicating a problem.

8.3.2 Why aren't all CPUs evenly being used on a shared micro partition?

On AIX 5.3 and later, the kernel scheduler dynamically increases and decreases the number of virtual processors based on the physical utilization of a shared micro partition. Increasing or decreasing the number of processors is also known as processor folding and is done by Virtual Processor Management.

For more information on Virtual Processor Management refer to the Performance Management Guide at:

http://pic.dhe.ibm.com/infocenter/aix/v7r1/index.jsp?topic=%2Fcom.ibm.aix.prftungd%2Fdoc%2Fprftungd%2Fvirtual_proc_mngmnt_part.htm

8.4 Memory Questions

8.4.1 How to tune VMM page replacement to reduce paging?

Please refer to chapter 8.2 How to tune VMM page replacement to reduce paging

8.4.2 When should I enable page_steal_method?

A good indication of when to enable page_steal_method is when performance monitoring tools like vmstat report a high scanned to freed ratio (sr/fr in vmstat). A high scanned to freed ratio typically occurs when only a small percentage of the real memory is being used for file caching.

Enabling page_steal_method requires a system reboot and therefore recommended to be enabled during a regular scheduled maintenance window.

Note: The page_steal_method is enabled by default on AIX 6.1.

8.4.3 What formula should I use to calculate minfree and maxfree?

There are several legacy formulas to calculate the values for minfree and maxfree based on the number of CPUs and the page read ahead tuning for the file systems. These formulas originate from the time prior AIX 5.3 which introduced new default values for minfree and maxfree.

The defaults for minfree and maxfree in AIX 5.3 and later are sufficient for most of the workloads and should only be changed in the case that a system is running low on free pages (fre column in vmstat).

When increasing minfree and maxfree is required to prevent a low free list, both tunables should be increased by the same value.

Note: It is not recommended changing minfree and maxfree to lower values than their default.

8.4.4 Memory pools are unbalanced; should memory affinity be turned off?

No. Memory affinity should not be disabled on systems that are on AIX 5.3 with bos.mp or bos.mp64 5.3.0.41 or later. Disabling memory affinity can introduce other performance problems since it also turns off other features that are important to performance.

Note: The sizes of the individual memory pools are derived from the information the Hypervisor provides AIX at boot time and cannot be altered explicitly.

8.4.5 Does 'avm' in vmstat indicate "available memory"?

No, avm in vmstat is the number of active virtual pages.

8.4.6 Why is my free memory so low?

One reason for low free memory is that memory is overcommitted, i.e. the amount of computational pages exceeds the amount of real memory available. This can be determined through lsps, and vmstat. A more common reason for low free memory is that AIX allocates pages from real memory for file pages until the amount of free memory goes below the minfree threshold at which page replacement gets started. This allows for caching of filesystem information, and is a performance benefit.

8.4.7 Should I pin my database memory using v_pinshm?

No. The appropriate and recommended way to assure that database memory is not being paged out to paging space is to follow the steps outlined in Section 8.2 recommended VMM page replacement tuning.

The potential risk involved by enabling v_pinshm is that the system runs short on memory pages and either hangs or crashes.

Exception: Large (16MB) and huge (16GB) pages require v_pinshm set to 1.

8.5 Disk I/O Questions

8.5.1 What is the recommended queue_depth for disks?

The queue_depth is an attribute of the disk or storage subsystem. Its possible values are exclusively defined by disk or storage device.

Please refer to the storage vendor for appropriate values for the queue_depth.

8.5.2 What is the recommended num_cmd_elems for FC adapters?

The num_cmd_elems attribute for Fiber Channel adapters defines the maximum number of commands that can be queued to the adapter. The maximum number of commands depends on the storage sub-system and should be set according to the storage vendors recommended values.

8.5.3 How to tune AIO on AIX 6.1?

AIX 6.1 has a new implementation of the AIO kernel extension which does not require any tuning for most workloads.

Note: It is not possible to apply AIX 5.3 tuning to the new AIO kernel extension since the aio0 device no longer exists.

8.5.4 How do I enable AIO on AIX 6.1?

AIO on AIX 6.1 is enabled by default but not shown as active until used.

8.5.5 What are good values for the file I/O pacing tunables minpout and maxpout?

In general, there are no good values for minpout and maxpout that would apply for all environments or workloads. Too small values could have a negative impact on file I/O performance and too high values don't serve the purpose of throttling the file I/O throughput.

If you decide to change the default values for minpout and maxpout please refer to the Disk I/O pacing section of the Performance Management Guide for good starting values:

http://pic.dhe.ibm.com/infocenter/aix/v7r1/index.jsp?topic=/com.ibm.aix.prftungd/doc/prftungd/disk_io_pacing.htm

8.5.6 Does CIO always results in better database performance?

No. Please refer to 8.7 How to migrate from cached file system to concurrent IO.

8.6 Network Questions

8.6.1 What are the recommended values for rfc1323, tcp_sendspace, and tcp_recvspace?

The TCP protocol has only a 16-bit value for the window size. In order to take advantage of a receivers tcp_recvspace value greater than 65536, you have to enable TCP window scaling. RFC1323 enables TCP window scaling so that the 16 bit value can be scaled up by powers of two to allow larger window size. That allows TCP to have more packets outstanding to fill either long latency network or higher speed networks. So anytime the tcp_recvspace is larger than 64K, you need RFC1323 enabled. For slower speed networks (like 10 or 100 Mbit Ethernet) its best not to enable this as it does add another 12 bytes to the TCP protocol header which takes away from user payload.

The recommended values for rfc1323, tcp_sendspace, and tcp_recvspace are the default values that are set through the interface specific network options, ISNO. The default values are set based on interface type and its configuration.

To display the interface specific network options for a specific interface run:

ifconfig <interface_name>

The Performance Management Guide has more information on the interface specific network options:

pic.dhe.ibm.com/infocenter/aix/v7r1/index.jsp?topic=/com.ibm.aix.prftungd/doc/prftungd/interface_network_opts.htm

8.6.2 When should I change rfc1323, tcp_sendspace, and tcp_recvspace?

The default values for rfc1323, tcp_sendspace, and tcp_recvspace set through interface specific network options are sufficient to get good network throughput performance for most environments. Changing their values to improve throughput performance is only required for connections with high latency. High latency typically occurs on wide area network connections.

Please note that using too large values for tcp_sendspace and tcp_recvspace can lead to a situation that network devices, such as switches, gets overrun and loses packets.

8.6.3 I want to change the rfc1323, tcp_sendspace and tcp_recvspace for multiple network interfaces; should I change them globally with the no command and turn off use_isno?

It is not recommended turning off use_isno for that purpose. Changes to network tunables that are available through interface specific network options should be changed for specific interfaces only. Permanent changes can be applied with chdev and dynamic changes with ifconfig.

8.6.4 What are dog threads?

Dog threads are dedicated kernel threads for network interfaces. When enabled for a network interface, all incoming packet will be queued to a kernel thread after the initial processing within the interrupt from the network device.

Enabling dog threads can improve throughput when a high speed network adapter bottlenecks on a single CPU. Enabling dog threads increases CPU utilization due to the dispatching overhead for the kernel threads and the additional queuing.

8.6.5 When should I enable dog threads for a network interface?

Most of the modern machines don’t need dog threads enabled for high speed adapters up through gigabit adapter speed. Receive intensive workloads on very high speed network adapters, like 10 gigabit Ethernet or InfiniBand, can improve the receive throughput on systems that have multiple CPUs.

Use ifconfig to enable dog threads dynamically or chdev to enable dog threads permanently. For more detailed information refer to the Performance Management Guide at:

pic.dhe.ibm.com/infocenter/aix/v7r1/index.jsp?topic=/com.ibm.aix.prftungd/doc/prftungd/enable_thread_usage_lan_adapters.htm

Using dog threads on large partitions:

With the large number of CPU hardware threads, the incoming packet workload can be spread (hashed) out to too many dog threads and that can limit the performance gains.

For example on a 32 thread CPU LPAR, you could limit the number of dog threads to 4 by doing "no -o ndogthreads= 4" (or add the -r option to enable on next reboot).

8.6.6 When should I enable link level flow control?

Link level flow control is a point-to-point protocol between two network devices; for example an AIX Ethernet adapter and a network switch. By default, AIX has link level flow control enabled for most of the supported network devices.

Note: Some network switches don't have link level flow control enabled by default.

8.6.7 What are checksum offload and large send?

High speed network adapters, like gigabit and 10 gigabit Ethernet, can compute the TCP checksum (checksum offload) for transmitting and receiving packets. Offloading the checksum computation reduces the CPU utilization on the host.

Large send allows the host to send TCP messages up to 64KB to the network adapter which re-segments the packet to MTU sized packets, computes their checksum and transmits them. It typically reduces the sending CPU utilization in half and can provide a significant performance improvement for 10 GigE Ethernet.

For more detailed information refer to the Performance Management Guide:

Checksum offload

pic.dhe.ibm.com/infocenter/aix/v7r1/index.jsp?topic=/com.ibm.aix.prftungd/doc/prftungd/tcp_checksum_offload.htm

Large send

pic.dhe.ibm.com/infocenter/aix/v7r1/index.jsp?topic=/com.ibm.aix.prftungd/doc/prftungd/tcp_large_send_offload.htm

8.6.8 When should I disable checksum offload and large send?

Checksum offload and large send should be disabled for debug purpose only when problem determination or performance analysis requires tracing of the IP/TCP layer.

There is one exception for large send. In rare cases large send can lead to a situation where a switch becomes saturated due to the high rate at which large packets are transmitted.

8.6.9 How can I improve the loopback performance?

TCP fast loopback option provides improved loopback performance by bypassing the TCP/IP stack and lo0 loopback driver and allows a shorter socket-to-socket path for improved performance. This is transparent to the application program as it uses the normal TCP/IP (AF_INET) sockets interface. This provides performance just slightly slower than AF_UNIX sockets, but without making any source code changes. Performance gains vary by workload but can be quite large and will also be more on larger SMP systems.

The TCP fast loopback can be enabled (on the fly for all future connections) by “no -o tcp_fastlo=1" (or add the -r option to enable on next reboot).

Because the TCP/IP and lo0 interface are bypassed (once the connection is established), the normal netstat -a and netstat -i statistics will not reflect the fast loopback packet traffic.

The TCP fast loopback option can be enabled for workload partitions (WPARs) by “no -o tcp_fastlo_crosswpar=1” (or add the -r option to enable on next reboot).

9 POWER7

9.1 Introduction

This chapter covers best practices for POWER7 performance.

9.2 Compatibility Mode

POWER7 supports partition mobility with POWER6 and POWER6+ systems by providing compatibility modes. Partitions running in POWER6 or POWER6+ compatibility mode can run in ST or SMT2. SMT4 and SIMD double-precision floating-point (VSX) are not available in compatibility mode.

9.3 Memory Considerations

Some workloads have memory per core requirements that exceed the amount of memory that a POWER7 system supports. The following options are available for workloads that require more memory per core:

- Active Memory™ Expansion (AME)
- Lower number of cores per socket (6-way POWER7)
- TurboCore™

9.4 Single thread versus SMT2/SMT4

Applications that are single process and single threaded may benefit from running in ST mode while multithreaded and/or multi process applications typically benefit more running in SMT2 or SMT4 mode. ST mode can be beneficial in the case of a multi process application where the number of application processes is smaller than the number of cores assigned to the partition. Applications that do not scale with a larger number of CPUs may also benefit from running in SMT2 or ST mode instead of SMT4 since lower number of SMT threads means lower number of logical CPUs.

ST, SMT2 or SMT4 mode can be set through the smtctl command. The default mode is SMT4.

9.5 Adapter Placement

High speed network adapters should be placed following the recommendations of the PCI Adapter Placement recommendations for the individual POWER7 model.

For detailed information refer to the POWER7 section of the System Hardware information at:

http://pic.dhe.ibm.com/infocenter/powersys/v3r1m5/index.jsp?topic=/p7hdx/power_systems.htm

9.6 Affinitized partitions

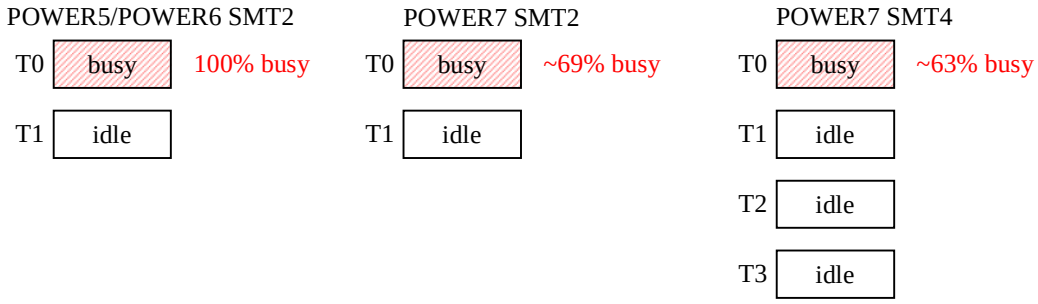
The POWER7 hypervisor was improved to maximize partition performance through affinitization. It optimizes the assignment of CPU and memory to partitions based on system topology. This results in a balanced configuration when running multiple partitions on a system.

When a partition gets activated, the hypervisor will allocate CPUs as close as possible to where allocated memory is located in order to reduce remote memory access. For shared partitions the hypervisor assigns a home node domain, the chip where the partition's memory is located, for each virtual processor. The hypervisor dispatches the shared partition's virtual processor(s) to run on the home node domain whenever possible. If dispatching on the home node domain is not possible due to physical processor over commitment of the system, the hypervisor will dispatch the virtual processor temporarily on another chip.

9.7 POWER7 CPU utilization reporting

Because of the advances with symmetric multiprocessing, legacy tools that display CPU usage are often no longer accurate. POWER7 introduces an improved reporting of the consumed capacity of a processor. This section explains the difference in CPU utilization reporting between POWER5, POWER6 and POWER7.

The figure below illustrates how CPU utilization is reported on POWER5, POWER6 and POWER7. On POWER5 and POWER6, when one of the two hardware threads in SMT2 mode is busy (T0) while the other one is idle (T1), the utilization of the processor is 100%. On POWER7, the utilization of the processor in SMT2 mode is around 69%, providing a better view about how much capacity is available.

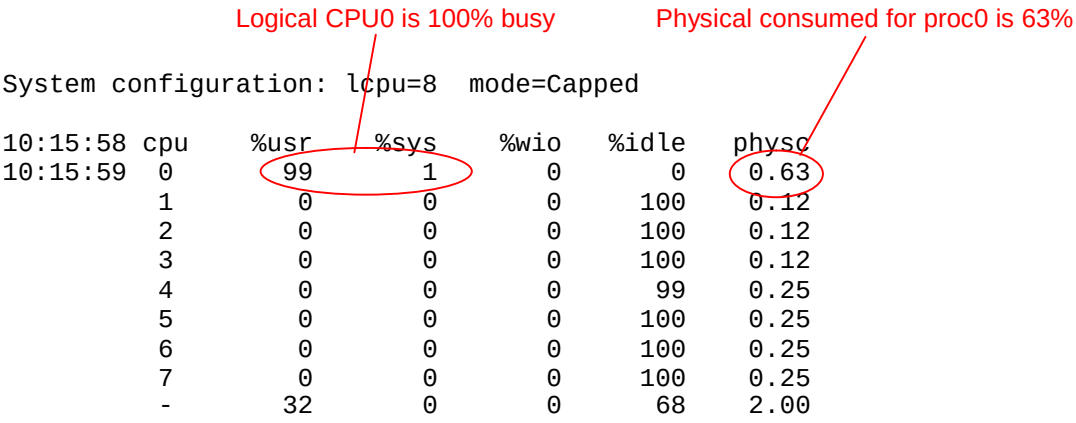


In SMT4 mode, with one hardware thread busy (T0) and the other three idle (T1, T2, and T3), the utilization of the processor is around 63%. The processor's utilization in SMT4 mode is less than in SMT2 mode since it has more capacity available through the additional two hardware threads.

9.7.1 CPU utilization example for dedicated LPAR

The following examples demonstrate the CPU utilization for a single threaded program running in SMT4, SMT2 and ST mode on a dedicated LPAR with two cores.

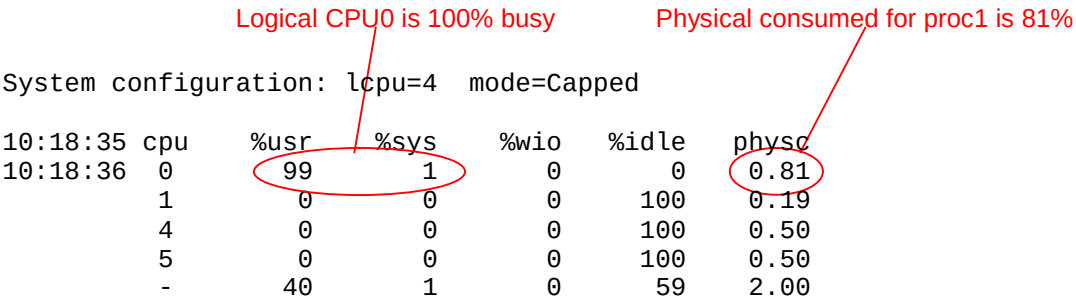
The first example below demonstrates the CPU utilization, as reported by the sar command, when running a single threaded application in SMT4 mode:



The example above shows that the single threaded program consumed an entire logical CPU (cpu0) but not the entire capacity of the processor.

Switching from SMT4 to SMT2 mode reduces the number of hardware threads that are available to execute code. Thus the consumed capacity when running the same single threaded program will be higher than in SMT4 mode.

The sar output below demonstrates the CPU utilization of the same single threaded program now running in SMT2 mode:



The single threaded program is now running on cpu0. Like in the SMT4 example, the program is consuming an entire logical CPU but now it is consuming 81% of the processor's capacity.

Switching to ST mode will cause the single threaded program to consume the entire capacity of a processor since there are no other hardware threads available to execute code.

Logical CPU0 is 100% busy

```
System configuration: lcpu=2  mode=Capped

10:22:02 cpu    %usr    %sys    %wio   %idle
10:22:03  0       99       1       0       0
          4        0       0       0     100
          -       50       0       0      50
```

Note: Switching modes from SMT4 to SMT2 and SMT2 to ST were done dynamically with smtctl for the examples above.

9.7.2 CPU utilization example for shared LPAR

The following examples demonstrate the CPU utilization for a single threaded program running in SMT4, SMT2 and ST mode on a dedicated LPAR with two virtual processors and an entitled capacity of 2.0.

The first example below is running the same workload as in the dedicated examples. It demonstrates the CPU utilization, as reported by the sar command, when running a single threaded application in SMT4 mode:

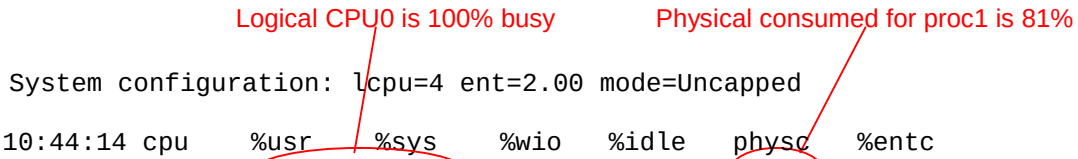
Logical CPU0 is 100% busy Physical consumed for proc0 is 63%

```
System configuration: lcpu=8 ent=2.00 mode=Uncapped

10:43:15 cpu    %usr    %sys    %wio   %idle   physc   %entc
10:43:16  0       99       1       0       0    0.63    31.7
          1        0       0       0     100    0.12     6.1
          2        0       0       0     100    0.12     6.1
          3        0       0       0     100    0.12     6.1
          4       17      34       0      48    0.00     0.1
          5        0       1       0      99    0.00     0.1
          6        0       1       0      99    0.00     0.1
          7        0       3       0      97    0.00     0.1
          U        -       -       0      50    0.99    49.7
          -       32       0       0      68    1.01    50.3
```

The example above shows that the single threaded program consumed an entire POWER7 SMT thread (logical CPU 0). The percentage entitlement consumed for this core was 50, with 31.7% for logical CPU 0 and 6.1% for CPU 1 to 3. This means that the LPAR received 100% capacity of a POWER7 core from the hypervisor; however, the workload utilized only one of the four SMT threads.

The sar output below demonstrates the CPU utilization of the same single threaded program after switching from SMT4 to SMT2 mode:

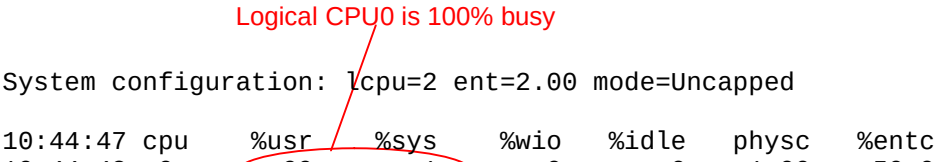



```
System configuration: lcpu=4 ent=2.00 mode=Uncapped

10:44:14 cpu    %usr    %sys    %wio   %idle   physc   %entc
10:44:15  0      99       1       0       0    0.81    40.6
          1       0       0       0     100    0.19     9.4
          4      29      48       0      22    0.00     0.1
          5       0       4       0      96    0.00     0.1
          U       -       -       0      50    1.00    49.9
          -      40       0       0      59    1.00    50.1
```

Like in the previous example, the LPAR received 100% capacity of a POWER7 core but utilizes only one of the two SMT threads.

Switching to ST mode will cause the single threaded program to consume the entire capacity of a processor since there are no other hardware threads available to execute code.




```
System configuration: lcpu=2 ent=2.00 mode=Uncapped

10:44:47 cpu    %usr    %sys    %wio   %idle   physc   %entc
10:44:48  0      99       1       0       0    1.00    50.0
          4      29      51       0      20    0.00     0.1
          U       -       -       0      50    1.00    49.9
          -      50       0       0      50    1.00    50.1
```

Note: Like in the dedicated examples, switching modes from SMT4 to SMT2 and SMT2 to ST was done dynamically with smtctl for the examples above.

9.8 AIX Scheduling

The AIX scheduler is optimized to provide best raw throughput and response time on POWER7 and POWER7+ based systems. To achieve this dispatching preferentially utilizes the primary SMT thread of each core with the net effect of spreading out workload across as many cores as needed/available.

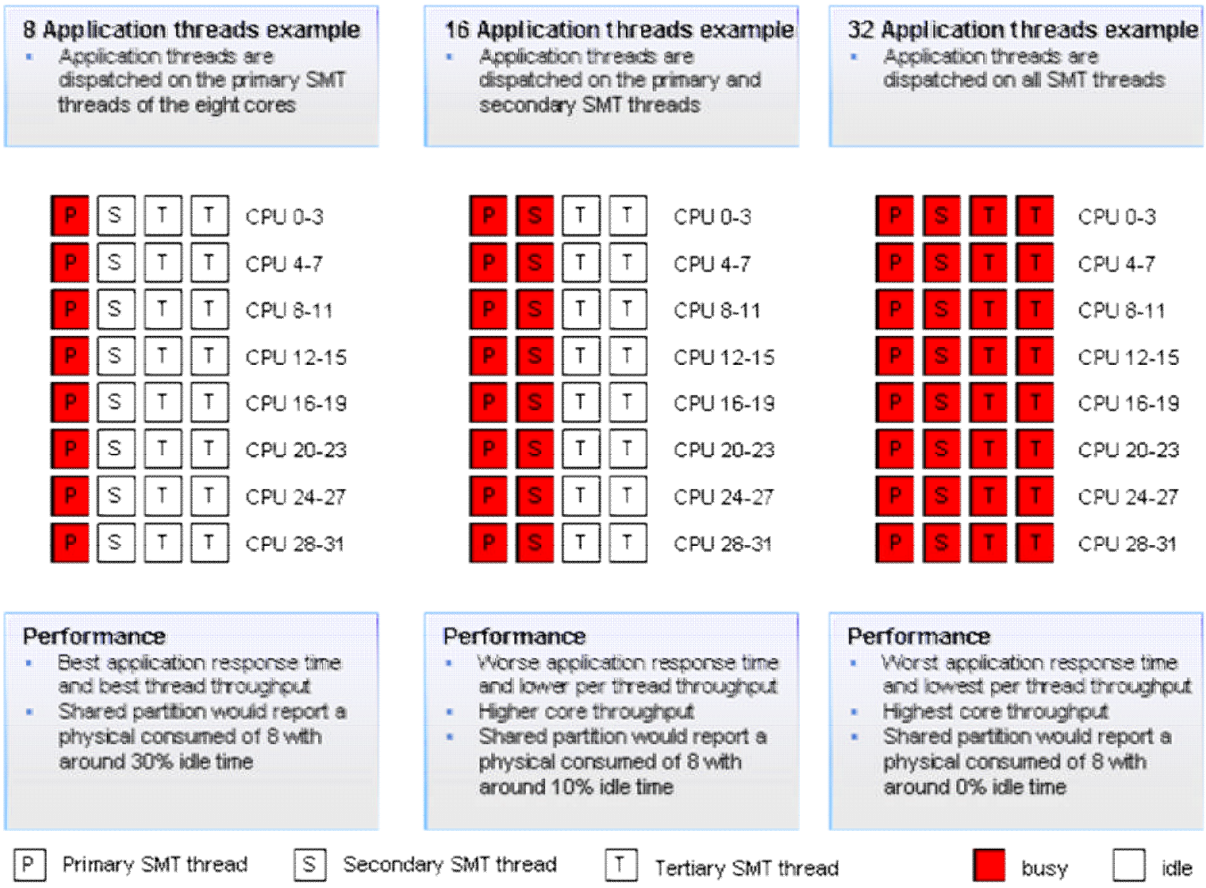
The following table is a simplified illustration of the SMT scheduling on an LPAR with 8 dedicated POWER7 cores or 8 virtual processors (VCPUs) and a compute intensive workload:

# Application threads	SMT scheduling	Mode
8	Primary SMT thread of all eight cores	ST
16	Primary and secondary SMT threads of all eight cores	SMT2
24	Primary, secondary and tertiary SMT threads of all eight cores	SMT4

32	Primary, secondary and tertiary SMT threads of all eight cores	SMT4

Note: The mode column represents the dynamic SMT mode set through the Intelligent Threads feature.

The following examples illustrate the workload distribution across the SMT threads when running eight, 16 or 32 application threads on a partition with 8 virtual processors that are running in SMT4 mode.



Note: Idle time reported for a shared processor partition also depends on its entitled capacity. The example above assumes an entitled capacity of 8.0 or less and eight virtual processors.

9.8.1 Scaled Throughput Dispatching

AIX 6.1 TL8 and AIX 7.1 TL2 introduce a new scaled throughput Virtual Processor Management (VPM) feature. This feature has four folding modes that controls on how the dispatcher places the workload onto the SMT threads of the cores or virtual processors. The folding mode can be change dynamically through the new schedo tunable vpm_throughput_mode.

The default mode (vpm_throughput_mode=0) is raw throughput mode. The dispatching of the workload in this mode is done as described in the previous section.

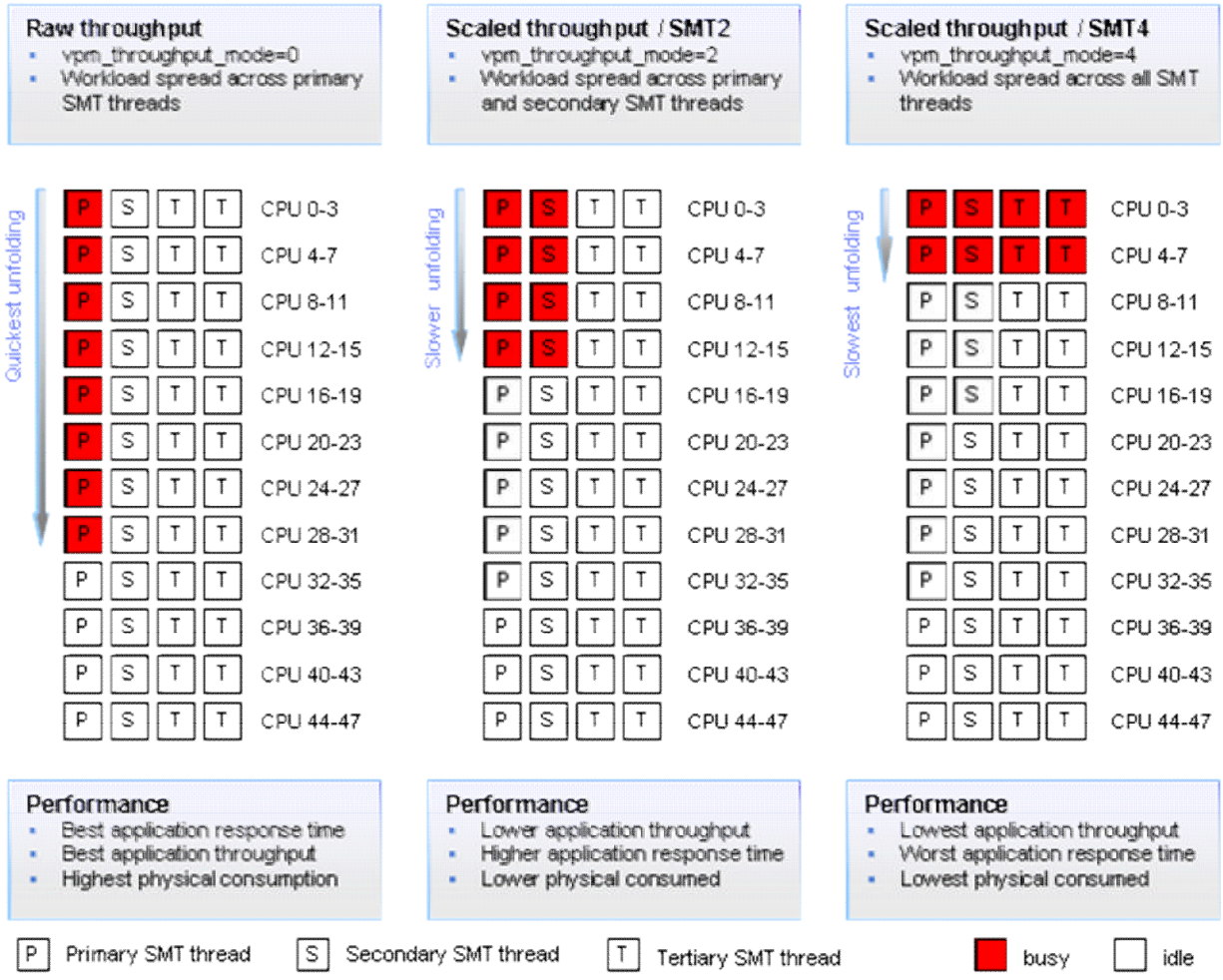
The enhanced raw throughput mode (vpm_throughput_mode=1) works similar to the default mode with the difference that it considers load as well as the utilization in folding decision. Depending on the workload, running in this mode may show lower physical CPU consumption than mode 0.

In scaled throughput mode SMT2 (vpm_throughput_mode=2) the dispatcher spreads the workload across the primary and secondary SMT threads of a virtual processor instead of spreading them across primary SMT threads only. Running in this mode results in a lower physical CPU consumption compared to raw or enhanced raw throughput mode.

In scaled throughput mode SMT4 (vpm_throughput_mode=4) the dispatcher spreads the workload across all four SMT threads of a virtual processor. This mode provides the lowest physical CPU consumption but also the highest application response time and lowest per application thread throughput.

The scaled throughput modes, vpm_throughput_mode 2 and 4, are a trade off between physical CPU consumption and application response time and throughput.

The following example illustrates the concept of raw and scaled throughput modes. In this example the workload consists of eight compute intensive application threads that are running on a partition with 12 virtual processors.



In raw throughput mode (vpm_throughput_mode=0) the eight application threads are spread across the primary SMT threads of eight virtual processor. This mode provides best application response time and application throughput. However, it also has the highest physical CPU consumption.

In scaled throughput SMT2 mode (vpm_throughput_mode=2) the eight application threads are spread across the primary and secondary SMT threads of four virtual processors. The physical CPU consumption is half of the raw throughput mode and per core throughput is higher. However, application response time will be longer and per application thread throughput will be lower.

In scaled throughput SMT4 mode (vpm_throughput_mode=4) the application threads are spread across all SMT threads of the virtual processor. This mode provides the lowest physical CPU consumption and the highest core throughput but also the worst application response time and per application thread throughput.

Note: The difference in application response time and throughput as well as the physical CPU consumption between the raw and scaled throughput mode may vary with the workload.

9.10 Virtualization Best Practices

This section summarizes the POWER7 Virtualization Best Practices. For detailed description and latest updates, please refer to the latest version of the POWER7 Virtualization Best Practice Guide at URL:
https://www.ibm.com/developerworks/wikis/download/attachments/53871915/P7_virtualization_bestpractice.doc?version=1

9.10.1 Sizing virtual processors

- CEC: the number of virtual processors of an individual LPAR should not exceed the number of physical cores in the system
- Shared processor pool: the number of virtual processors of an individual LPAR should not exceed the number of physical cores in the shared processor pool

9.10.2 Entitlement considerations

Best practice for LPAR entitlement would be to set the LPARs entitlement capacity to its average physical CPU usage and let the peaks addressed by additional uncapped cycles. For example, an LPAR running a workload that has an average physical consumed of 3.5 cores and a peak utilization of 4.5 cores should have 5 virtual processors to handle the peak CPU usage and an entitlement of 3.5.

9.10.3 Virtual Processor Management

Virtual Processor Management, also known as processor folding, controls how many virtual processors of an LPAR are enabled for processing. Once a second AIX determines how many virtual processors are required based on the aggregate CPU utilization. In the case that the current aggregate CPU utilization is above folding threshold, vpm_fold_threshold, an additional virtual processor will be enabled. In the case that the aggregate CPU utilization is below the folding threshold, a virtual processor will be disabled.

Prior to AIX 6.1 TL6 the vpm_fold_threshold represented the core utilization and had a threshold of 70%. Starting with AIX 6.1 TL6 the vpm_fold_threshold represents the SMT thread utilization and the new threshold value is 49%. It is important to understand that the effective threshold has not changed.

Note: Changing the vpm_fold_threshold value is not recommended. The vpm_fold_threshold tunable therefore is a restricted tunable.

9.10.4 Best memory and CPU resource assignment for critical LPARs

A good way to ensure that a critical LPAR get the best resources assigned is to activate the critical LPAR before activating any other LPAR, including VIO server LPAR(s). In the case that

the critical LPAR depends on a VIO server LPAR, it will not boot the operating system until the VIO server has been activated.

PowerVM Firmware 730 introduced support for affinity groups allowing to group LPARs to be placed within a single or few domains. Domains depend on the hardware topology.

9.11 Virtual Ethernet Performance

Virtual Ethernet enables IP based communication between logical partitions on the same system without going through physical Ethernet devices. This section explains tuning methods that can help to improve the performance over virtual Ethernet.

9.11.1 Sending large data packets through "largesend"

The largesend feature allows sending large data packets over virtual Ethernet adapters without breaking up the packets into smaller, MTU size packets

Starting with AIX 6.1 TL7 SP1 and AIX 7.1 SP1 the operating systems that supports the mtu_bypass attribute for the shared Ethernet adapter provide a persistent way to enable the largesend feature.

To determine if the operating system supports the mtu_bypass attribute run the following lsattr command:

```
lsattr -El enX |grep by_pass
```

If the mtu_bypass attribute is supported, the above command will return:

```
mtu_bypass    off             Enable/Disable largesend for
virtual Ethernet True
```

Enable largesend through:

```
chdev -l enX -a mtu_bypass=on
```

If the mtu_bypass attribute isn't available, use the non-persistent method to enable largesend:

```
ifconfig enX largesend
```

Add the above command is a rc file to re-enable largesend on reboot.

Regardless of the method used to enable largesend, it is recommended to verify that largesend actually is enabled by running:

```
ifconfig enX
```

Example output:

```
en2:
flags=1e080863,4c0<UP,BROADCAST,NOTRAILERS,RUNNING,SIMPLEX,MULTICAST,G
ROUPRT,64BIT,CHECKSUM_OFFLOAD(ACTIVE),LARGESEND,CHAIN>
        inet 192.1.1.2 netmask 0xffffff00 broadcast 192.1.1.255
         tcp_sendspace 262144 tcp_recvspace 262144 rfc1323 1
```

Note: largesend needs to be enabled on all client partitions

9.11.2 Virtual Ethernet Adapter Buffers

A virtual Ethernet adapter uses buffers of different sizes for the data communication through virtual Ethernet. The default number of buffers typically is sufficient for most environments. Environments with a heavy network load over virtual Ethernet may experience a buffer shortage which leads to dropped packets at the Hypervisor level. Dropped packets usually results in packet retransmission and slower network throughput.

The following example is the Hypervisor send and receive failure statistic reported by “entstat –d entX”:

```
Hypervisor Send Failures: 0
  Receiver Failures: 0
  Send Errors: 0
Hypervisor Receive Failures: 0
```

The Hypervisor increments the “Hypervisor Send Failures” counter every time it cannot send a packet due to a virtual Ethernet adapter buffer shortage. It also increments either the “Receiver Failure” or the “Send Errors” counter depending on where the buffer shortage occurred. The “Receiver Failure” gets incremented in the case the LPAR to which the packet should be send had no buffer available to receive the data. The “Send Error” gets incremented in the case that the sending LPAR is short on buffers.

The Hypervisor increments the “Hypervisor Receive Failures” counter in the case that another LPAR couldn’t send data to us because we had no buffer available.

It’s important to understand that the Hypervisor always increments the failure counters on both LPAR if the data couldn’t be received due to a buffer shortage on the target LPAR.

For example: LPAR “A” wants to send a packet to LPAR “B” but LPAR “B” has no buffer available to receive the packet.

The packet will be dropped and the Hypervisor increments the “Hypervisor Send Failure” and “Receiver Failures” on LPAR “A”

```
Hypervisor Send Failures: 1
  Receiver Failures: 1
  Send Errors: 0
Hypervisor Receive Failures: 0
```

The Hypervisor increments the “Hypervisor Receive Failure on LPAR “B”:

```
Hypervisor Send Failures: 0
  Receiver Failures: 0
  Send Errors: 0
Hypervisor Receive Failures: 1
```

A buffer shortage can have multiple reasons. One reason would be that the LPAR does not get sufficient CPU resources because the system is heavily utilized or the LPAR significantly over commits its entitlement. Another possibility would be that the number of virtual Ethernet buffers currently allocated might be too small for the amount of network traffic through the virtual Ethernet. The following example output from the "entstat –d entX" command demonstrates the current and historical allocation of virtual Ethernet buffers as well as the amount of memory allocated for these buffers:

```
Receive Information
  Receive Buffers
    Buffer Type              Tiny    Small   Medium    Large     Huge
    Min Buffers               512      512      128       24       24
    Max Buffers              2048     2048      256       64       64
    Allocated                 512      512      128       24       24
    Registered                512      512      128       24       24
    History
      Max Allocated           512      699      128       24       24
      Lowest Registered       510      502      128       24       24
  Virtual Memory
    Minimum (KB)              256     1024     2048      768     1536
    Maximum (KB)             1024     4096     4096     2048     4096
  I/O Memory
    VRM Minimum (KB)          4096     4096     2560      864     1632
    VRM Desired (KB)         16384    16384     5120     2304     4352
    DMA Max Min (KB)         16384    16384     8192     4096     8192
```

In the above example, the "Max Allocated" for the "small" buffers is higher than its minimum value. This means that the demand on small buffer exceeded the minimum value and a dynamic buffer allocation took place. This alone does not indicate any problem. However, in the case that the Hypervisor reported receive failure the dynamic buffer allocation might not have been able to keep up with the demand. Increasing the minimum amount of the small buffer to a higher value than the maximum allocated would help to overcome this issue. In this example we would double the minimum number of small buffers from 512 to 1024.

To increase the minimum buffers with chdev, run

chdev -l entX -a min_buf_small=<number of buffers> -P

Example:

chdev -l ent0 -a min_buf_small=1024 -P

For systems with a heavy network load it is recommended setting the minimum buffers to the same value as the maximum buffers. This will prevent any dynamic buffer allocation and the LPAR will always have all buffers available.

Note: It is recommended to first check the CPU usage of the LPAR before making any virtual Ethernet buffer tuning.

9.11.3 Data Cache Block Flush

The Data Cache Block Flush feature is an attribute of the virtual Ethernet adapter. Enabling this feature allows the virtual Ethernet device driver to flush the data from the processors data cache after the data has been received. This can improve the virtual Ethernet performance in the case of a heavy network load with large packets transmitted over virtual Ethernet.

To enable the Data Cache Flush feature, run

```
chdev -l entX -a dcbflush_local=yes
```

Enabling the Data Cache Block Flush feature will increase CPU utilization since additional CPU cycles are needed to flush the data cache blocks. For most environments it provides the best benefit when enabled on the VIO clients.

9.11.4 Shared Ethernet Adapter

The Shared Ethernet Adapter serves as a bridge between physical and virtual Ethernet network. All tuning feature discussed in the previous sections do apply to its virtual Ethernet adapter. However, it is important to ensure that all clients LPARs on the virtual Ethernet support a feature like the largesend before making any tuning changes.

As a guideline, enabling largesend and large_receive on the Shared Ethernet Adapter should only be done if the Shared Ethernet Adapter serves AIX partitions only and none of the partitions is doing IP forwarding.

To enable largesend and large_receive on the VIOS Shared Ethernet Adapter, do one of the following:

During SEA creation, add the -attr at the end of the mkvdev command (skip /usr/ios/cli/ioscli if login padmin shell):

```
/usr/ios/cli/ioscli mkvdev -sea entX -vadapter entY -default
entY -defaultid Z -attr large_receive=yes largesend=1
```

To change the SEA attributes

```
/usr/ios/cli/ioscli chdev -dev entX -attr large_receive=yes
largesend=1
```

To display the SEA attributes

```
/usr/ios/cli/ioscli lsdev -dev entX -attr
```

Note: large_send and large_receive on the underlining physical adapter should be enabled by default. If they are not, enable them before setting up the SEA adapter.

9.12 Storage Virtualization Best Practice

9.12.1 VIOS Sizing and Uncapped Shared Weight

VIOS partitions are commonly deployed in pairs. The best CPU sizing methodology is to ensure that the guaranteed entitlement for the VIOS is sufficient for the average load, and that enough vCPUs are allocated to sustain its own peak load, and that of the corresponding VIOS pair. It is also vital to increase the uncapped shared weight capacity to be greater than all vSCSI client partitions to ensure that the VIOS has priority over the extra uncapped cycles it may require when I/O requests are re-routed to one VIOS during a failover event.

9.12.2 vSCSI client queue depth

The default queue depth for a virtual drive is only 3. This value was a safe setting at the time of implementation, but too restrictive for today's disk I/O devices, which are capable of much more I/O parallelism. For optimal performance, the virtual queue depth should match the capabilities of the hardware, meaning that it should be equal to the physical storage device's queue depth.

9.12.3 vSCSI virtual adapter count

Once the virtual drive queue depths are set, it is important to verify that enough virtual adapters are available to the vSCSI client to support the virtual drives and all of their outstanding I/Os. Each virtual adapter has 512 command elements; however, 2 are reserved for the adapter, and 3 for each virtual disk. So, the following formula can be used to determine the number of virtual drives that can be attached behind a vSCSI adapter.

virtual_drives = (512 -2) / (virtual_q_depth + 3)

In this example, if the virtual drives have a queue depth of 16, then 26 virtual drives can be supported by a single vSCSI adapter. (((512-2) / (16+3)) = 26.8)

9.13 Performance Advisor Tools

IBM STG Cross Platform Systems Performance team developed several performance advisor tools that can help to identify performance bottlenecks. These tools run with an AIX partition where they analyze the performance characteristics and provide a health check report.

The performance advisor tools are available at the following URLs:

PowerVM Virtualization Performance LPAR Advisor

https://www.ibm.com/developerworks/wikis/display/WikiPtype/PowerVM+Virtualization+performance+lpar+advisor

VIOS Advisor

http://www.ibm.com/developerworks/wikis/display/WikiPtype/VIOS+Advisor

Java Performance Advisor

https://www.ibm.com/developerworks/wikis/display/WikiPtype/Java+Performance+Advisor

10 Java

10.1 Introduction

This section describes the Java performance best practices for running Java on AIX.

10.2 32-bit versus 64-bit Java

Some processor architectures provide better performance with a 64-bit JVM (Java Virtural Machine) verses a 32-bit JVM due to a increased number of processor registers available in 64-bit mode. This requires running Java applications in a 64-bit JVM to achieve the best performance even though the application itself might not have any requirements to run in 64-bit mode.

The Power architecture does not require running Java applications in a 64-bit JVM to achieve the best performance since its 32-bit mode is not limited by a small number of processor registers.

For best performance, use a 32-bit JVM unless the memory requirement of the application requires a 64-bit environment.

10.3 Medium page size usage (64K pages)

Java applications can gain performance by using 64K page size for the Java native heap which can be enabled by setting the following loader control environment variable:

```
LDR_CNTRL=DATAPSIZE=64K@TEXTPSIZE=64K@STACKPSIZE=64K
```

In addition to the Java native heap the 64K page size for Java heap can provide additional performance improvements. This feature can be enabled by adding the following flag to the Java command options:

```
-Xlp64k
```

Note: Starting with IBM Java 6.0 SR7 64K page size is the default.

10.4 Application Scaling

Each SMT thread is represented as a logical CPU by the operating system. Thus, a partition running in SMT4 mode will have twice the number of logical CPUs as a partition that has the same number of cores but is running in SMT2 mode.

Applications that do not scale with the number of CPUs might not show the expected performance when running in SMT4 mode than in SMT2 mode. A way to mitigate this problem is to switch from SMT4 mode to SMT2 mode. This can be done with the `smtctl` command by running:

```
smtctl -t 2 -w boot
```

10.5 Enhanced Tuning with Resource Sets and Memory Affinity

An alternative way to address application scaling issues is the use of resource sets (rsets) which allows running an application on a limited number of CPUs. An easy way to create a resource set and start the application attached to it in one step is through the `execrset` command. The following example creates a resource set with CPU 0 to 3 and starts the application attached to it:

```
execrset -c 0-3 -e <application>
```

Enabling memory affinity can improve the application performance even more since the application is limited to run on a defined number of CPUs and memory can be allocated that is close to them. To enable memory affinity the MEMORY_AFFINITY environment variable needs to be set to MCM in the applications environment:

```
MEMORY_AFFINITY=MCM
```

Running multiple application instances and balancing the workload across them is a common way to overcome scaling issues. The following example demonstrates how to start multiple instances of an application with each application instance limited to run on an individual processor of a system running in SMT4 mode.

```
execrset -c 0-3 -e <application instance 1>
execrset -c 4-7 -e <application instance 2>
execrset -c 8-11 -e <application instance 3>
execrset -c 12-15 -e <application instance 4>
```

Application instance 1 will run on the four SMT threads, represented by logical CPU 0 to 3, of processor 0; application instance 2 will run on the four SMT threads of processor 1, and so on. Running the smtctl command without any flag will show which logical CPUs belongs to which processor.

Note: A user must have root authority or have CAP_NUMA_ATTACH capability to use rsets.

11 IBM AIX Dynamic System Optimizer

11.1 Introduction

This section describes IBM AIX Dynamic System Optimizer (DSO), a new feature that optimizes system performance in an autonomous fashion.

11.2 Overview

DSO is built on the Active System Optimizer (ASO) framework introduced in AIX 7.1 TL1 SP1. The ASO framework includes a user-space daemon namely “aso”, advanced instrumentation methods based on kernel data and the hardware Performance Monitoring Unit (PMU) and two optimizations for cache and memory affinity. ASO provides autonomous tuning by grouping workload threads to a set of cores close together and improving placement of frequently accessed pages.

DSO extends ASO to provide two additional optimizations: large page and memory prefetch. DSO identifies heavily used regions of memory, automatically upgrading pages that are benefited from page size promotion i.e. migrating 4K and 64K to 16MB page sizes. It also monitors memory access patterns and performs dynamic hardware DSCR (Dynamic Stream Control Register) tuning. DSO includes both process level DSCR and system wide dynamic DSCR support.

11.3 How to enable ASO/DSO

11.3.1 Prerequisites

- POWER7 or POWER7+
- AIX V6.1 TL8 SP1 (both ASO/DSO functionality) OR
- AIX V7.1 TL1 SP1 (ASO only) OR
- AIX V7.1 TL2 SP1 (both ASO/DSO)

11.3.2 Install and enable DSO

DSO is a separately priced offering. Customer must order DSO package, install and activate it. Once DSO is licensed, one would need to install the dso.aso fileset via AIX smitty or “installp” command. Use one of the following methods to turn enable ASO/DSO.

- “asoo –po aso_active=1” for system wide setting AND
- enable at process level via environment variables: ASO_ENABLED or ASO_OPTIONS

ASO_ENABLED=<ALWAYS/NEVER>

ASO_OPTIONS=<ALL=ON/OFF | CACHE_AFFINITY=ON/OFF | MEMORY_AFFINITY=ON/OFF | LARGE_PAGE=ON/OFF | MEMORY_PREFETCH=ON/OFF>

Example:

$ export ASO_OPTIONS=” ALL=OFF, CACHE_AFFINITY=ON”

In this example, only cache optimization is turned on, all other optimizations, memory affinity, large page, memory prefetch are off.

11.4 Performance Expectation

The goal of DSO is to achieve optimal system performance autonomously. Once enabled, DSO routinely analyzes workload in real time and automatically adjusts settings to make efficient use of system resources. DSO performance benefit varies dependent on the workload characteristic and configuration. Below is some quick guidelines.

- Dedicated vs. SPLPAR

DSO produces performance improvement for both dedicated and Shared Processor LPAR (SPLPAR) environment.

- DSO Overhead

DSO overhead is expected to be within 3% or less of processor resource. On shared processor partition systems, this will correspond to 3% of the defined CPU capacity. Laboratory results have showed that DSO incurs very minimal performance overhead for a variety of workloads.

- Cache and memory affinity optimization

Multi-threaded workloads are benefited from cache and memory affinity. Java, WebSphere, and DB2 workloads have showed double digit performance gain in laboratory environment.

- Large page and memory prefetch optimization

Long running, large DB workloads are excellent candidate for large page and memory prefetch optimization. To take advantage of these two optimizations, the CPU utilization for the workload must be at least 2 cores for large pages and 8 cores for memory prefetch optimization. The memory requirement for the workload must be at least 16GB of System V shared memory for both optimizations.

11.5 Reference

For further reading on DSO, refer to the following document.

- POWER7 AND POWER7+ Optimization and Tuning Guide
http://www.redbooks.ibm.com/redpieces/abstracts/sg248079.html?Open&pdfbookmark

- IBM AIX Dynamic System Optimizer
http://pic.dhe.ibm.com/infocenter/aix/v7r1/topic/com.ibm.aix.optimize/optimize_pdf.pdf

12 Reporting a Performance Problem

12.1 Introduction

The focus on this chapter is on what information and data should be provided when reporting a performance problem to post-sales support. However, the information provided in this chapter also applies when reporting performance problems to pre-sales organizations.

12.2 Define the Performance Problem

The first question support will ask when a performance problem is reported is “What is slow?” A good understanding about what exactly is slow helps the performance analyst concentrate the analysis on the component(s) that is most likely causing the performance problem.

Let’s take the example of an application server that is running slower than usual when accessed through a site network, while it is performing well when accessed locally or through a private network. Without the information that the problem only exists on the site network, the performance analyst most likely would not start the analysis with looking into the network components that connects the application server to the site network.

As a general rule, the better the description of the performance problem the more likely is a quick identification of the root cause and potential resolution.

A good starting point to describe a performance problem is to answer the questions in the PROBLEM.INFO file that is shipped with the performance data collection tool PERFPMR.

12.3 Performance Data Collection using PERFPMR

The most commonly used tool to collect system performance data on AIX is the performance data collection tool PERFPMR. PERFPMR is a package of tools that collect performance data along with software and hardware configuration information that is required for performance analysis.

The goal of PERFPMR is to collect a good base of information that can be used by performance analysts to get started with performance analysis and problem resolution. The process of collecting performance data may be repeated multiple times if

- the initial data did not capture the problem
- additional data is required after tuning changes have been applied
- the data collection itself needs to be adjusted to capture sufficient data

The PERFPMR data collection consists of three steps:

1. Kernel trace data collection

PERFPMR collects two kernel traces. The first kernel trace is a “full” trace which has most of the trace hooks enabled. The second trace runs with lock instrumentation enabled and collects lock events.

2. Monitoring

System monitoring data, such as the outputs of sar, vmstat, iostat, are collected for the time duration specified for the data collection. In addition, statistic data for network, VMM, etc are collected before and after the monitor data collection.

3. Configuration data collection

Detailed hardware and software configuration is collected when the performance data collection has finished.

For most cases, the kernel trace data collection is the most important part of the entire data collection process since the kernel traces provide the most detailed information about the kernel activity. Therefore it's important that the performance problem occurs at the time the PERPMR data collection is started.

PERFPMR is available for each supported version of AIX. It can be downloaded from URL:

ftp://ftp.software.ibm.com/aix/tools/perftools/perfpmr/

Each PERFPMR package contains a README file which describes the installation of PERFPMR, how to collect performance data and where to upload the collected data.

12.4 PERFPMR vs. test case

Some performance problems can be recreated by running a command or a simple program that performs a specific operation that is slow. A performance analyst can use such a test case to recreate the performance problem on a test system and perform root cause analysis without impacting a client's production environment.

When providing a simplified test case to demonstrate a performance problem it is important that it represents the real problem. A slow running command or simplified test case may or may not have the same root cause as a slow running system or application.

Providing a simplified test case and PERFPMR data taken while the system or application was running slow will allow the performance analyst to assure that the right root cause gets addressed.

12.5 Questions that help IBM diagnose the problem

The following is a list of questions that help to define a performance problem. The same list of questions is shipped with PERFPMR as PROBLEM.INFO file. It is recommended to answer the question in the PROBLEM.INFO file when reporting a performance problem.

- Can you append more detail on the simplest, repeatable example of the problem?
  - Can the problem be demonstrated with the execution of a specific command or sequence of events?
  - 'ls /slow/fs' takes 60 seconds or

    - Binary mode 'ftp' put from one specific client only runs at 20 Kbytes/second.
    - etc.
  - If not, describe the least complex example of the problem.
  - Is the execution of AIX commands also slow?
- Is this problem a case of something that had worked previously (ie. before a upgrade) and now does not run properly?
  If so:
  - Describe any other recent changes?
    I.e. workload, number of users, networks, configuration, etc.
- Or is this a case of an application/system/hardware that is being set up for the first time?
  If so:
  - What performance is expected?
  - What is the expectation based on?
- Is the slow performance intermittent?
  - Is there any pattern to the slow behavior?
  - Does it get slow, but then disappear for a while?
  - Does it get slow at certain times of day or relation to some specific activity?
  - About how long is the period of slow performance before it returns to normal?
  - Is it slow when the system is otherwise idle?
    (i.e. capacity vs. elapsed time)
  - What is the CPU utilization when the system is idle after a period of slow performance (use 'vmstat 1')?
    (perhaps something is looping)
- Are all applications/commands/users slow or just some?
- What aspect is slow?
  i.e.
  - Time to echo a character
  - Elapsed time to complete the transaction
- Does rebooting the system make the problem disappear for a while?
  (i.e. a resource may be consumed but not freed back up)
  - If so, about how long until the problem reappears?
- If client/server, can the problem be demonstrated when run locally on the server (network vs. server issue)?
- Does the problem disappear when the application is run from the system console?
- If client/server, from the client how long does a 'ping server_ip_address' take?
  (use the server_ip_address to exclude nameserver and other variables. i.e. 'ping 129.35.33.22')
- If network related, please describe the network segments including bandwidth (i.e. 10mbit/sec, 9600 baud...) and routers between the client and server.

- What vendor applications are on the system and are they involved in the performance issue?
- What is the version/release/level of the vendor applications?
- Have they been updated recently?

IBM Corporation
Systems and Technology Group
Route 100
Somers, New York 10589

Produced in the United States of America
February 2010

POW03049-USEN-03